APPLE II®

TEKNINEN KÄSIKIRJA



apple computer inc.

HUOM!

Apple Computer Inc. pidättää itsellään oikeuden tehdä muutoksia tässä käsikirjassa esiteltyihin tuotteisiin ilman erillistä ilmoitusta.

10260 Bandley Drive
Cupertino, CA 95041
(408) 996-1010

Apple numero A2L0001A (030-0004-01)

Kirjoittanut Christopher Espinosa

Apple II Tekninen Käsikirja

TEKNINEN KÄSIKIRJA
APPLE II
JA APPLE II PLUS
PIENTIETOKONEITA VARTEN

LUKU 1
YLEISKUVAUS

LUKU 2
KESKUSTELU APPLEN KANSSA

LUKU 3
SYSTEM MONITOR

LUKU 4
MUISTIN RAKENNE

LUKU 5
INPUT/OUTPUT-JÄRJESTELMÄT

LUKU 6
LAITTEISTON RAKENNE

SISÄLLYSLUETTELO

JOHDANTO

Tämä on Apple II ja Apple II Plus -pientietokoneita varten laadittu tekninen käsikirja. Kuten Apple itse, myös tämä kirja on työkalu ja siihen pätee sama kuin muihinkin työkaluihin: on hyvä tietää hiukan ennen kuin alkaa käyttää sitä.

Tämä kirja ei ole ohjelmoinnin oppikirja. Se sisältää pelkkiä tietoja, ei lainkaan keinoja. Jos olet juuri purkanut laitteesi paketista tai et vielä hallitse yhtään siihen sopivaa ohjelmointikieltä, Sinun on syytä laskea tämä kirja syrjään ja tutustua johonkin muuhun Applen käsikirjaan. Riippuen siitä, minkälaisen Applen olet ostanut, Sinulla tulisi olla jompi kumpi seuraavista:

Apple II BASIC Programming Manual
(Apple numero A2L0005)

Applesoft Tutorial
(Apple numero A2L0018)

Nämä ovat Applen käyttämän BASIC-kielen version oppikirjoja. Ne sisältävät myös täydelliset ohjeet Applen käyttökuntoon saattamiseksi. Tämän kirjan lopussa oleva kirjallisuusluettelo sisältää joukon kirjoja, jotka saattavat kiinnostaa Sinua.

Applesta on olemassa joitakin versioita ja tämä käsikirja sopii niihin kaikkiin. On mahdollista, että kaikkia esitettyjä ominaisuuksia ei ole juuri Sinun Applessasi. Kun kirjassa esitetään tietoja, jotka eivät sovellu kaikkiin Appleihin, varoitetaan siitä alaviitteellä.

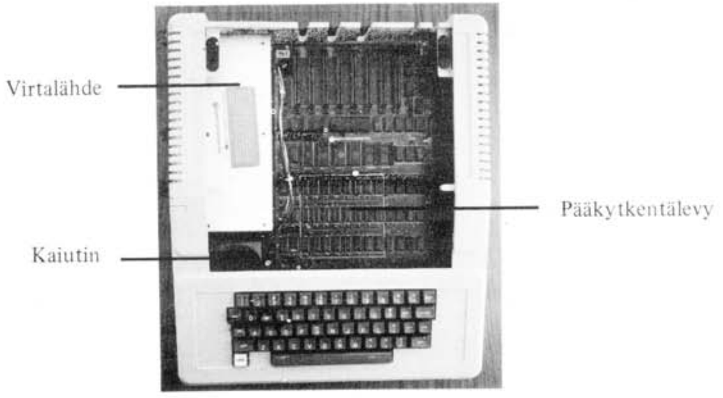
Tämä käsikirja käsittelee Apple II tietokonetta, sen osia ja toimintoja. Kirjassa on erilliset luvut System Monitorista, syöttö/tulostusvälineistä ja niiden toiminnasta sekä itse Applen elektronisesta rakenteesta. Jos tarvitset tietoa jostakin muusta Apple-tuotteesta, tutustu sen mukana seuraavaan ohjekirjaan.

LUKU 1
YLEISKUVAUS

Yksityiskohtaista tietoa Applen saattamisesta käyttökuntoon on kirjojen Apple BASIC Programming Manual ja Applesoft Tutorial ensimmäisessä luvussa.

Tässä käsikirjassa kaikki suuntaa osoittavat ilmaukset on määritelty seuraavasti: Kun Apple on edessäsi oikein päin (näppäimistö sinua kohti), »eteen» ja »alas» ovat näppäimistöön päin, »taakse» ja »ylös» ovat päinvastaiseen suuntaan. Irroita Applen kansi nostamalla sitä takareunasta, kunnes kuuluu napsahdus. Vedä tämän jälkeen kantta suoraan taaksepäin ja nosta se pois.

Tältä laite näyttää:

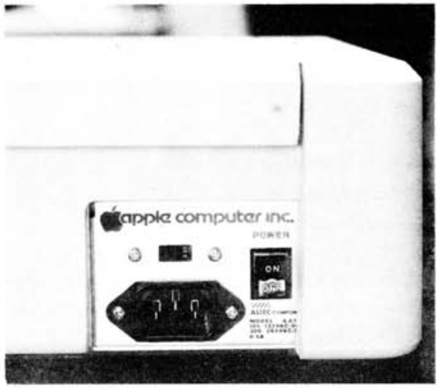


VIRTALÄHDE

Vasemmassa laidassa oleva metallilaatikko on virtalähde. Se syöttää neljää jännitettä: + 5V, - 5,2V, +11,8V ja - 12,0V. Se on korkeataajuinen hakkurityyppinen virtalähde, johon on suunniteltu useita suojauksia estämään eri jännitteiden välinen epätasapaino. Tietokoneen verkkojohto tulee suoraan virtalähteen takaosaan. Päävirtakatkaisin on myös virtalähteessä. Tämän tarkoituksena on pitää huoli siitä, että sinä ja sormesi eivät vahingossa päädy suljetun korkeajännitepiirin osasiksi.



110 voltin malli



110/220 voltin malli

PÄÄKYTKENTÄLEVY

Suuri vihreä painokytkentälevy, joka muodostaa suurimman osan laitteen pohjasta on tietokone itse. Apple II:n kytkentälevystä on kaksi hieman erilaista versiota: alkuperäinen (Versio 0) ja versio 1. Levyjen väliset pienet erot piilevät elektroniikassa ja niihin palataan jatkuvasti eri kappaleissa. Yhteenveto eroista on kappaleessa »Applen variaatioita» sivulla 25.

Applen kytkentälevyssä on noin 80 integroitua piiriä ja kourallinen muita komponentteja. Levyn keskellä, heti takana olevien kahdeksan »kultahampaisen» oheislaiteliitännän edessä on muita suurempi integroitu piiri. Se on Applen aivot, Synertek/MOS Technology:n mikroprosessori 6502. Applessa se suorittaa noin 1.023.000 konejaksoa sekunnissa ja voi tehdä yli viisisataatuhatta yhteen- tai vähennyslaskua samassa ajassa. Sen osoitevaraus on 65.536 kahdeksan bitin tavua. 6502:n käskyvalikoimaan kuuluu 56 käskyä ja 13 osoitusmuotoa. Tätä mikroprosessoria ja muita sen versioita käytetään myös monissa muissa tietokoneissa ja elektronisissa laitteissa.

Aivan mikroprosessorin alapuolella on kuusi IC-kantaa, jotka voivat sisältää 1—6 hieman pienempiä integroituja piirejä. Nämä piirit muodostavat Applen lukumuistin (ROM). Niissä ovat Applen ohjelmat, jotka ovat käytettävissäsi heti kun virta kytketään päälle. Näitä ovat esimerkiksi Apple System Monitor, Apple Autostart Monitor, Apple Integer BASIC ja Applesoft II BASIC sekä Apple Programmer's Aid #1 alirutiinipaketti. ROM-piirien määrä ja sisältö riippuu Applen tyypistä ja hankkimistasi lisälaitteista.

Aivan ROM-piirien ja kytkentälevyn keskellä olevan kiinnitysruuvin alapuolella on valkoisella neliöllä rajattu alue, joka sisältää 24 integroitujen piirien liitoskantaa. Osassa tai kaikissa niistä voi olla integroituja piirejä. Nämä ovat Applen hajasaantimuistin (RAM) perusyksiköt. Applessa voi olla 4.092—49.152 tavua RAM-muistia.* Jokaisessa rivissä voi olla kahdeksan integroitua piiriä, joiden kapasiteetti on joko 4K tai 16K. Kussakin rivissä täytyy olla kahdeksan samantyyppistä

* Voit laajentaa laitteistosi RAM-muistin määrää yhteensä 64 kilotavuun hankkimalla Apple Kielikortin (Apple Language Card). Tämä kortti (tuote n:o A2B00006) kuuluu osana Applen kielijärjestelmään.

muistikomponenttia. Muistityyppejä voidaan kuitenkin haluaessa yhdistellä riveittäin, niin että lopputuloksena voidaan saada yhdeksän eri muistikokoa*. RAM-muistiin varastoidaan kaikki ohjelmat ja data, jota tarvitaan ohjelman aikana. RAM-muistiin varastoitu tieto häviää, kun virta katkaistaan.

Muilla emolevyn komponenteilla on erilaisia tehtäviä: ne valvovat koneen osasta toiseen kulkevaa informaatiovirtaa, keräävät tietoja ulkopuolisesta maailmasta tai lähettävät niitä käyttämällä näyttöä tai kaiutinta.

Levyn takaosassa olevasta kahdeksasta oheislaitteen väyläliitännästä jokainen voi sisältää yhden oheislaitekortin, joka sallii RAM- tai ROM-muistin laajentamisen tai liittää Applen kirjoittimeen tai muuhun syöttö/tulostuslaitteeseen.

KESKUSTELU APPLEN KANSSA

Applelle puhutaan sormien välityksellä. Useimmat käytetyt ohjelmat ja kielet odottavat niille puhuttavan näppäimistön kautta. Se on, lukuunottamatta muutamia erityisjärjestelyitä ja lisänäppäimiä, kuin tavallinen kirjoituskoneen näppäimistö. Saadaksesi nopean yleiskuvan näppäimistöstä, käy läpi sivut 6—12 Apple II BASIC Programming Manualista tai sivut 5—11 Applesoft Tutorialista.

Koska puhuminen suoritetaan sormilla, on luonnollista, että kuuntelu tapahtuu silmien välityksellä. Apple kertoo, mitä se on tekemässä tulostamalla näyttöön kirjaimia, numeroita, symboleja ja joskus värillisiä ruutuja ja viivoja.

---

* Apple II on suunniteltu niin, että siinä voidaan käyttää sekä 16K:n että 4K:n RAM-muisteja. 16K:n piirien hyvän saatavuuden ja laskeneiden valmistuskustannusten johdosta toimitetaan nykyään vain 16K:n piirejä.

APPLEN NÄPPÄIMISTÖ

Applen näppäimistö (alkuperäinen)

Näppäinten lukumäärä:	52
Koodaus:	Isot ASCII standardin mukaiset kirjaimet
Koodien määrä:	91
Ulostulo:	Seitsemän bittiä, plus strobe
Tehontarve:	+5V/120mA −12V/50mA
Rollover:	2 näppäintä
Erikoisnäppäimet:	CTRL ESC RESET REPT ← →

Muistiin sijoittuminen:

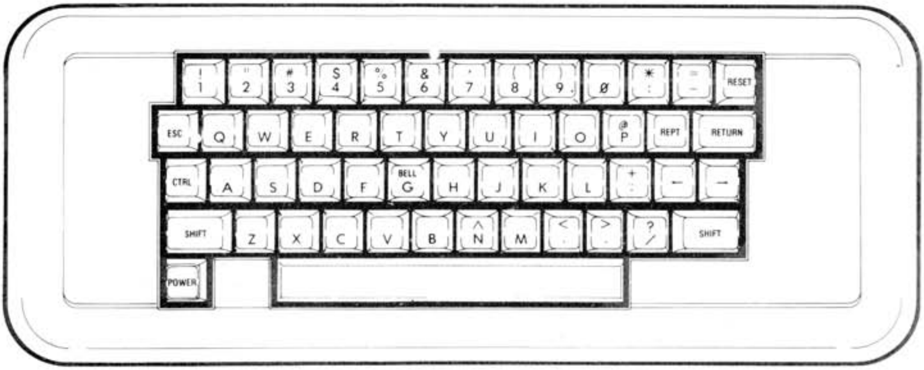
	Hex	Desim.	
Data	\$C000	49152	-16384
Nollaus	\$C010	49168	-16368

Applessa on 52:n merkin näppäimistö, joka käyttää American Standard Code for Information Interchange (ASCII)*-koodia. 91 ASCII-merkkiä 96:sta voidaan aikaansaada tällä näppäimistöllä. Taulukko 2 esittää Applen näppäimet ja niitä vastaavat ASCII-koodit. Kuva 3 taas esittää itse näppäimistöä.

Näppäimistö on sähköisesti yhdistetty kytkentälevyyn 16 johdinta sisältävällä litteällä kaapelilla. Johdinten päässä on pistokkeet, jotka yhdistävät ne niille varattuihin integroitujen piirien kantoihin. Kaapelin toinen pää on yhdistetty näppäimistöön, toinen taas pistokkeella emolevyn näppäimistöliittimeen. Tämä liitin on aivan emolevyn etureunassa näppäimistön itsensä alla. Liittimien kytkentä selviää tämän kirjan sivulta 102.

Useimmat Applen käyttämät ohjelmointikielet sisältävät käskyjä, joilla ohjelmaan voidaan syöttää näppäimistöltä tietoa nopeasti ja helposti (esim. INPUT- ja GET-käskyt BASICissa). Ohjelmat voivat kuitenkin lukea näppäimistöä myös suoraan.

* Kaikissa Applen käyttämissä ASCII-koodeissa on normaalisti ylin bitti asetettuna. Tämä on standardin mukainen merkki-pariteettinen ASCII-koodi (Mark-Parity ASCII).



Valokuva 3. Applen näppäimistö

NÄPPÄIMISTÖN LUKEMINEN

Näppäimistö lähettää tietoa yhdellä kertaa seitsemän bittiä, jotka yhdessä muodostavat yhden merkin. Nämä seitsemän bittiä – ja signaali, joka osoittaa, milloin näppäintä on painettu – muodostavat yhden muistipaikan sisällön ja ovat siinä muodossa ohjelman käytettävissä. Ohjelmat saavat selville näppäimistön tilan lukemalla tämän muistipaikan sisällön. Kun näppäintä painetaan, ko. muistipaikan arvoksi tulee 128 tai suurempi. Tämä arvo on kirjoitetun merkin numeerinen koodi. Taulukossa 3 sivulla 8 ovat ASCII-merkit ja niiden koodit. Muistipaikka säilyttää saamansa arvon, kunnes toista näppäintä painetaan tai ohjelma käskee muistipaikkaa unohtamaan senhetkisen sisältönsä.

Kun ohjelma on vastaanottanut ja hyväksynyt näppäimen painalluksen, sen pitäisi kehottaa näppäimistölle varattua muistipaikkaa »vapauttamaan» sisältämänsä merkki ja valmistautumaan uuden vastaanottamiseen. Ohjelma voi tehdä tämän viittaamalla toiseen muistipaikkaa, jolloin edellisen arvo laskee alle 128:n. Tämä arvo säilyy, kunnes uutta näppäintä painetaan. Tämä toimenpide on nimeltään »näppäimistön STROBEN nollaus». Ohjelma voi tehdä tämän joko lukemalla tietyn muistipaikan sisällön tai kirjoittamalla sinne. Luetulla tai kirjoitetulla tiedolla ei ole merkitystä. Pelkkä muistipaikan maininta suorittaa ko. toimenpiteen. Tämänkin jälkeen viimeksi painetun merkin koodi voidaan palauttaa lisäämällä 128 (heksadesimaalisena $80) näppäimistölle varatun muistipaikan arvoon.

Näppäimistö käyttää seuraavia muistipaikkoja:

Taulukko 1: Näppäimistön erikoismuistipaikat

Muistipaikka Hex	Desim.		Kuvaus
$CØØØ	49152	-16384	Näppäimistön data
$CØ1Ø	49168	-16368	Näppäimistön Stroben nollaus.

RESET -näppäin oikeassa yläkulmassa ei saa aikaan ASCII-koodia, vaan se on yhdistetty suoraan mikroprosessoriin. Kun tätä näppäintä painetaan, kaikki prosessorin toiminta lakkaa. Kun näppäin päästetään ylös, kone aloittaa nollausjakson (RESET CYCLE). Tarkemmat tiedot tästä tapahtumasta löytyvät sivulta 36.

CTRL ja SHIFT -näppäimet eivät nekään itse kehitä koodia, vaan pelkästään muuttavat muiden näppäinten tuottamia koodeja.

REPT -näppäin — jos sitä painetaan yksin — tuottaa viimeksi kirjoitetun merkin kaksoiskappaleen. Jos REPT -näppäintä painetaan ja pidetään yhtaikaa sitä sekä jotain muuta näppäintä alhaalla, on seurauksena sama kuin jos jälkimmäistä näppäintä painettaisiin katkuvasti 10 kertaa sekunnissa. Tämä toistotoiminta loppuu heti kun jompikumpi näppäin päästetään ylös.

POWER-valo alhaalla vasemmalla on merkkilamppu, joka osoittaa, milloin Applen virta on päällä.

Taulukko 2. Näppäimet ja niitä vastaavat ASCII-koodit

Näppäin	Yksin	CTRL	SHIFT	Molemmat	Näppäin	Yksin	CTRL	SHIFT	Molemmat
space	$AØ	$AØ	$AØ	$AØ	RETURN	$8D	$8D	$8D	$8D
Ø	$BØ	$BØ	$BØ	$BØ	G	$C7	$87	$C7	$87
1!	$B1	$B1	$A1	$A1	H	$C8	$88	$C8	$88
2"	$B2	$B2	$A2	$A2	I	$C9	$89	$C9	$89
3#	$B3	$B3	$A3	$A3	J	$CA	$8A	$CA	$8A
4$	$B4	$B4	$A4	$A4	K	$CB	$8B	$CB	$8B
5%	$B5	$B5	$A5	$A5	L	$CC	$8C	$CC	$8C
6&	$B6	$B6	$A6	$A6	M	$CD	$8D	$DD	$9D
7'	$B7	$B7	$A7	$A7	N^	$CE	$8E	$DE	$9E
8(	$B8	$B8	$A8	$A8	O	$CF	$8F	$CF	$8F
9)	$B9	$B9	$A9	$A9	P@	$DØ	$9Ø	$CØ	$8Ø
:*	$BA	$BA	$AA	$AA	Q	$D1	$91	$D1	$91
;+	$BB	$BB	$AB	$AB	R	$D2	$92	$D2	$92
,<	$AC	$AC	$BC	$BC	S	$D3	$93	$D3	$93
-=	$AD	$AD	$BD	$BD	T	$D4	$94	$D4	$94
.>	$AE	$AE	$BE	$BE	U	$D5	$95	$D5	$95
/?	$AF	$AF	$BF	$BF	V	$D6	$96	$D6	$96
A	$C1	$81	$C1	$81	W	$D7	$97	$D7	$97
B	$C2	$82	$C2	$82	X	$D8	$98	$D8	$98
C	$C3	$83	$C3	$83	Y	$D9	$99	$D9	$99
D	$C4	$84	$C4	$84	Z	$DA	$9A	$DA	$9A
E	$C5	$85	$C5	$85	→	$88	$88	$88	$88
F	$C6	$86	$C6	$86	←	$95	$95	$95	$95
					ESC	$9B	$9B	$9B	$9B

Kaikki koodit on annettu hexadesimaalisina. Löytääksesi vastaavat desimaaliarvot, käytä taulukkoa 3.

Taulukko 3: ASCII-merkkivalikoima									
Desimaali:		128	144	16Ø	176	192	2Ø8	224	24Ø
	Hex:	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
Ø	$Ø	nul	dle		Ø	@	P		p
1	$1	soh	dc1	!	1	A	Q	a	q
2	$2	stx	dc2	"	2	B	R	b	r
3	$3	etx	dc3	#	3	C	S	c	s
4	$4	eot	dc4	$	4	D	T	d	t
5	$5	enq	nak	%	5	E	U	e	u
6	$6	ack	syn	&	6	F	V	f	v
7	$7	bel	etb	'	7	G	W	g	w
8	$8	bs	can	(	8	H	X	h	x
9	$9	ht	em	)	9	I	Y	i	y
1Ø	$A	lf	sub	*	:	J	Z	j	z
11	$B	vt	esc	+	;	K	[	k	{
12	$C	ff	fs	,	<	L	\	l	|
13	$D	cr	gs	–	=	M	]	m	}
14	$E	so	rs	.	>	N	^	n	~
15	$F	si	us	/	?	O	_	o	rub

Taulukossa esiintyvät kahden ja kolmen pienen kirjaimen ryhmät ovat lyhennyksiä ASCII-standardin kontrollimerkeistä.

Kaikkia taulukossa lueteltuja merkkejä ei saada aikaan Applen näppäimistöllä. Erikoisesti merkit, jotka ovat kahdessa äärimmäisenä oikealla olevassa sarakkeessa (pienet kirjaimet), symbolit [(vasen hakasulku), \ (vinoviiva) ja kontrollimerkit »fs», »us» ja »rub» eivät kuulu Applen näppäimistöön.

Jokaista taulussa olevaa merkkiä vastaavat desimaali- ja heksadesimaaliluvut saadaan laskemalla yhteen sen rivin ja sarakkeen luvut, joilla ao. merkki on.

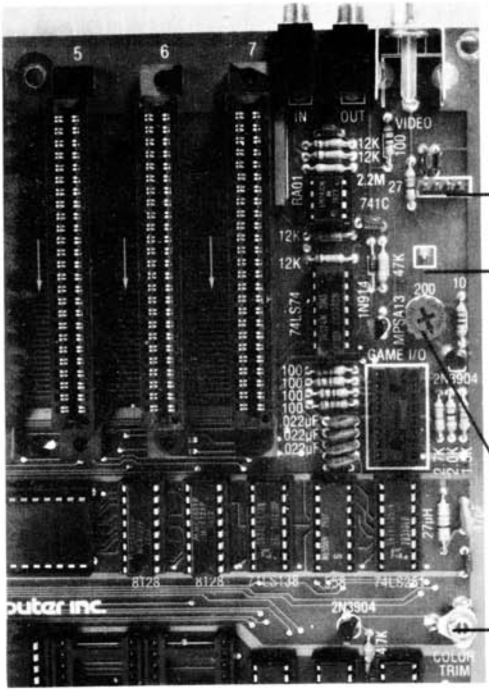
Applen videonäyttö	
Näytön tyyppi:	Käyttää osaa järjestelmän muistiavaruudesta (Memory Mapped)
Näyttötavat:	Teksti, perusgrafiikka, tarkkuusgrafiikka
Tekstitulostus:	960 merkkiä (24 riviä joilla 40 merkkiä/rivi)
Merkin rakenne:	5 x 7 pisteen matriisi
Merkkivalikoima:	Isot kirjaimet ASCII-standardin mukaisesti, 64 kirjainta.
Tulostuksen tilat:	Normaali, käänteinen ja vilkkuva
Grafiikkakapasiteetti:	1.920 ruutua (perusgrafiikka) 40 x 48 ruudun matriisissa. 53.760 pistettä (tarkkuusgrafiikka) 280 x 192 pisteen matriisissa
Värien määrä:	16 (perusgrafiikka) 6 (tarkkuusgrafiikka)

VIDEOLIITÄNTÄ

Apple II levyn oikeassa takanurkassa on metallinen liitin, jonka yläpuolella lukee »VIDEO». Tähän liittimeen voit kytkeä kaapelin Applen ja suljetussa piirissä olevan videomonitorin välille. Kaapelin Applen puoleisessa päässä tulisi olla ulospuolinen »RCA»-liitin ja toisessa ao. monitoriin sopiva liitin. Liittimestä saatava signaali on Electronic Industries Association (EIA) -standardin mukainen ja National Television Standards Committee (NTSC) -sovellutuskelpoinen. Se on positiivinen värillinen videosignaali. Signaalin taso voidaan sovittaa nollan ja yhden voltin välille oikeassa laidassa n. kahdeksan senttimetrin päässä takareunasta olevalla pienellä pyöreällä potentiometrillä.

Ei-säädettävä kahden voltin versio samasta signaalista on saatavissa kahdesta muusta paikasta: erillisestä WIRE-WRAP -liittimestä* (kuva 4) levyn vasemmassa laidassa viitisen senttiä takareunasta, sekä yhdessä neljästä samanlaisesta liittimestä vasemmassa laidassa lähellä levyn takareunaa. Kolme muuta liitintä on kytketty -5 voltin jännitteeseen, + 12 voltin jännitteeseen ja maahan. Sivulla 97 on täydellinen kuvaus tästä ylimääräisestä videoliitännästä.

* Tätä nastaa ei ole Version Ø Apple II -järjestelmän levyissä.



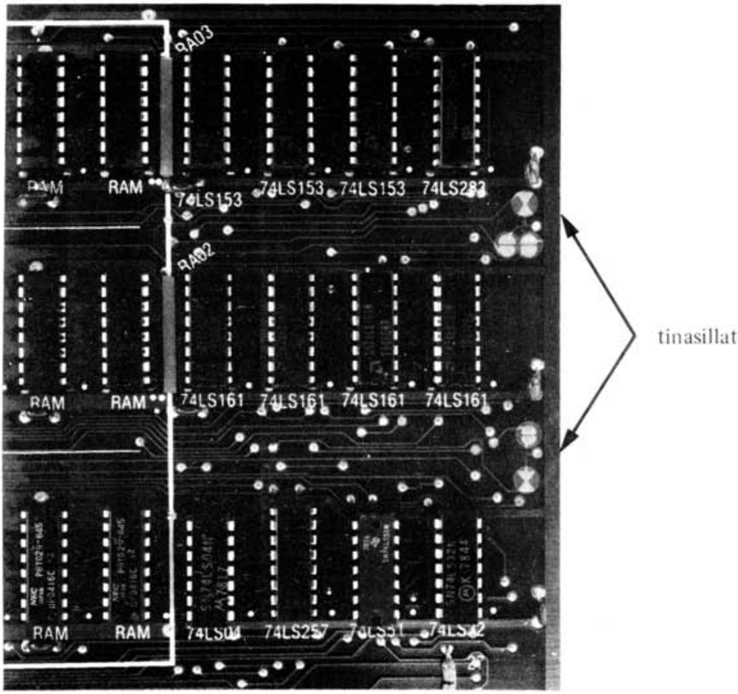
Valokuva 4. Applen videoliittimet ja potentiometri

EURAPPLE (50 Hz) MUUNNOS

Apple voi olla muunnettu siten, että se tuottaa monien Euroopan maiden käyttämän CCIR-standardin mukaisen videosignaalin. Tämä muunnos suoritetaan katkaisemalla kaksi X:n muotoista foliosiltaa, jotka ovat oikeassa laidassa n. yhdeksän tuumaa levyn takareunasta) ja juottamalla yhteen kolme O:n muotoista foliosiltaa (jotka sijaitsevat suunnilleen samalla kohdalla). Tarkemmat paikat selviävät kuvasta 5. tämän muutoksen jälkeen Apple voidaan liittää eurooppalaisten standardien mukaiseen mustavalko- tai värimonitoriin. Tarvittaessa voidaan hankkia myös Eurocolor-muunnin, joka muuttaa viedosignaalin PAL- tai SECAM-standardien mukaiseksi televisiosignaaliksi. Tämä on pieni levy, joka liitetään Applen oikeanpuoleisimpaan oheislaiteliitäntään (väylä 7) ja yhdistetään omalla liitosjohdollaan ylimääräiseen videoulostuloliittimeen.

NÄYTTÖ

Applen näyttöä voidaan käyttää kolmenlaisen informaation esittämiseen:



Valokuva 5. Eurapplen (50 hz) tinasillat

1) **Teksti.** Applen näyttöön mahtuu 24 riviä numeroita, erikoismerkkejä ja isoja kirjaimia. Joka riville mahtuu 40 merkkiä. Merkit muodostetaan pistematriisilla, joka on seitsemän pistettä korkea ja viisi pistettä leveä. Jokaisen merkin ja rivin välissä on yhden pisteen levyinen tyhjä väli.

2) **Perusgrafiikka.** Näytössä voi olla 1.920 värillistä ruutua, jotka muodostavat 40 ruutua leveän ja 48 ruutua korkean matriisin. Jokaisen ruudun väri voidaan valita 16 vaihtoehdosta. Ruutujen välissä ei ole tyhjää tilaa, joten kaksi samanväristä rinnakkaista ruutua näyttää yhdeltä isommalta.

3) **Tarkkuusgrafiikka.** Applella voidaan esittää myös värillinen pistematriisi, joka on 280 merkkiä leveä ja 192 pistettä korkea. Nämä pisteet ovat samankokoisia kuin ne, joilla kirjaimet muodostetaan. Tarkkuusgrafiikassa on valittavana kuusi väriä: musta, valkoinen, punainen, sininen, vihreä ja violetti.* Jokainen piste ei voi kuitenkaan olla mitä väriä tahansa.

Kun Apple esittää tietyntyyppistä informaatiota, sen sanotaan olevan asianomaisessa tilassa. Jos näytössä on siis numeroita ja kirjaimia, on mitä todenäköisintä, että Apple on tekstitilassa. Jos näyttö taas on täynnä värikkäitä ruutuja, laite on perusgrafiikkatilassa. Näytön alareunassa voi, kummassa grafiikkatilassa tahansa, myös olla neljän rivin korkuinen »otsikkoteksti». Perusgrafii-

* Version Ø Applen levyissä on 4 väriä: musta, valkoinen, vihreä ja violetti.

kassa nämä neljä riviä korvaavat kahdeksan alinta ruuturiviä, joten jäljelle jää 40 x 40 matriisi. Tarkkuusgrafiikassa korvautuu 32 alinta pisteriviä niin, että jäljelle jää 280 x 160 -matriisi. Näillä sekatiloilla voidaan esittää tekstiä ja grafiikkaa yhtaikaa, mutta molempia grafiikkoja ei kuitenkaan ole mahdollista käyttää samalla kertaa.

NÄYTTÖMUISTI

Näyttö käyttää systeemin RAM-muistia saadakseen selville, mitä sen on tulostettava. Yhden muistipaikan sisältö määrää näytössä olevan tietyn kohdan, joka voi olla merkki, kaksi päällekkäistä ruutua tai seitsemän vierekkäisen pisteen rivi. Teksti- ja perusgrafiikkatilojen tarvitsema informaatio on sijoitettu 1024 muistipaikan alueelle. Teksti ja perusgrafiikka käyttävät tätä samaa muistialuetta. Tarkkuusgrafiikka tarvitsee erillisen, suuremman (8192 muisipaikkaa) alueen, koska esitettävän informaation määrä on suurempi. Näitä muistialueita kutsutaan tavallisesti sivuiksi. Tarkkuusgrafiikalle varattua aluetta taas kutsutaan usein kuvapuskuriksi, koska sinne tavallisesti sijoitetaan kuva tai piirros.

NÄYTTÖMUISTIN SIVUT

On itse asiassa kaksi aluetta, joilta kukin tila voi ottaa tarvitsemansa informaation. Ensimmäisestä alueesta käytetään nimityksiä »ensimmäinen sivu» tai »sivu 1» ja toisesta vastaavasti »toinen sivu» tai »sivu 2». Sivuja vastaavat muistialueet ovat samanakokoisia ja välittömästi peräkkäin. toinen sivu on kätevä, kun halutaan varastoida kuva tai teksti, joka pitäisi pystyä esittämään heti tarvittaessa. Ohjelma voi tuottaa animaatiota näillä kahdella sivulla: piirtämällä toisen kun toinen on vielä näytössä tai vaihtamalla niitä nopeasti peräkkäin.

Teksti ja perusgrafiikka jakavat toisen sivun muistialueen aivan kuten ensimmäisenkin sivun. Molemmat sekatilat ovat käytössä myös toisella sivulla, mutta sivuja ei voi sekoittaa keskenään.

Taulukko 4: Videonäytön käyttämät muistialueet

Näyttö	Sivu	Alkukohta Hex	Alkukohta Desimaali	Loppukohta Hex	Loppukohta Desimaali
Teksti/ perusgrafiikka	Sivu 1	$400	1024	$7FF	2047
	Sivu 2	$800	2048	$BFF	3071
Tarkkuus-grafiikka	Sivu 1	$2000	8192	$3FFF	16383
	Sivu 2	$4000	16384	$5FFF	24575

NÄYTTÖKYTKIMET

Niitä kytkimiä, jotka hoitavat valinnan eri tilojen välillä, kutsutaan »pehmeiksi» kytkimiksi. Ne ovat kytkimiä, koska niillä on kaksi asentoa (esim. on/ei, teksti/grafiikka) ja ne ovat »pehmeitä», koska tietokoneen varusohjelmisto valvoo niitä. Ohjelma voi muuttaa kytkimen asentoa

viittaamalla sitä vastaavaan muistipaikkaan. Muistipaikkaan kirjoitetulla tai sieltä luetulla tiedolla ei ole merkitystä. Pelkkä viittaus kyseisen muistipaikan osoitteeseen muuttaa kytkimen asentoa.

Näytön »pehmeitä» kytkimiä valvoo kahdeksan muistipaikkaa. Ne toimivat pareittain: Kun parin toiseen muistipaikkaan viitataan, sitä vastaava tila kytketään »päälle» ja sen pari »pois». Nämä parit ovat

Taulukko 5: Näytön ohjelmalliset kytkimet

Muistipaikka Hex	Desim.		Kuvaus
$C050	49232	-16304	Näyttöön jokin Grafiikkatila.
$C051	49233	-16303	Näyttö tekstille.
$C052	49234	-16302	Näyttö kokonaan tekstille tai grafiikalle.
$C053	49235	-16301	Sekoitettu teksti ja grafiikka. *
$C054	49236	-16300	Näytä perussivu (sivu 1)
$C055	49237	-16299	Näytä apusivu (sivu 2)
$C056	49238	-16298	Näyttöön perusgrafiikka. *
$C057	49239	-16297	Näyttöön tarkkuusgrafiikka. *

Kytkimistä saadaan kymmenen erilaista yhdistelmää:

Taulukko 6: Näytön yhdistelmämahdollisuudet

Perussivu (sivu 1)			Apusivu (sivu 2)		
Näyttö	Kytkimet		Näyttö	Kytkimet	
Teksti	$C054	$C051		$C055	$C051
Kokonaan perus-grafiikalle	$C054 $C052	$C056 $C050	Kokonaan perus-grafiikalle	$C055 $C052	$C056 $C050
Kokonaan tarkkuus-grafiikalle	$C054 $C052	$C057 $C050	Kokonaan tarkkuus-grafiikalle	$C055 $C052	$C057 $C050
Sekoitettu teksti ja perusgrafiikka	$C054 $C053	$C056 $C050	Sekoitettu teksti ja tarkkuusgrafiikka	$C055 $C053	$C056 $C050
Sekoitettu teksti ja perusgrafiikka	$C054 $C053	$C057 $C050	Sekoitettu teksti ja tarkkuusgrafiikka	$C055 $C053	$C057 $C050

(Ne teistä, jotka ovat tutustuneet binääriaritmetiikkaan, kysyvät tietysti heti : »Missä muut kuusi kytkintä ovat?» – neljästä kaksisuunntaisesta kytkimestä saadaan kuusitoista erilaista yhdistelmää – vastaus piilee TEKSTI/GRAFIIKKA -kytkimessä. Kun kone on tekstitilassa, se voi olla samalla jossain kuudesta perusgrafiikka/tarkkuusgrafiikka -yhdistelmässä, sekatilassa tai sivunvalinnassa. Mutta koska Apple näyttää tekstiä, nämä eri grafiikat ovat näkymättömissä.)

Jotta Apple saataisiin johonkin näistä tiloista, ohjelman on vain viitattava ko. kytkinten asentoa vastaavan muistipaikan osoitteeseen. Konekieliset ohjelmat käyttävät yllämainittuja heksadesimaaliosoitteita; BASIC-ohjelmien taas – PEEK tai POKE käskyissä – niiden desimaalivastineita, jotka löytyvät taulukosta 5. Kytkimet voidaan asettaa missä järjestyksessä tahansa. Jompaan kumpaan grafiikkatilaan siirryttäessä on kuitenkin viisainta kääntää TEKSTI/GRAFIIKKA -kytkin viimeisenä. Kaikki muut muutokset pysyvät tällöin näkymättöminä tekstin takana, joten kun grafiikkatila astuu voimaan, grafiikkanäyttö saadaan valmiina.

* Nämä tilat ovat näkyvinä ainoastaan kun näyttökytkin on asetettu näyttämään grafiikkaa.

TEKSTITILA

Tekstitilassa Apple voi esittää 24 riviä, jokaisella rivillä 40 merkkiä. Jokainen merkki edustaa yhtä näytössä olevan sivun muistipaikan sisältöä. Merkkivalikoima sisältää 26 isoa kirjainta, 10 numeroa ja 28 erikoismerkkiä – yhteensä 64 merkkiä. Merkit muodostetaan 5 pistettä leveällä ja 7 pistettä korkealla pistematriisilla. Jokaisen merkin ja rivin välissä on yhden pisteen levyinen tyhjä väli. Merkit muodostetaan tavallisesti valkeilla pisteillä tummaa taustaa vasten. Jokainen merkki voidaan kuitenkin esittää myös tummina pisteinä valkealla taustalla tai vuorotellen näiden kahden yhdistelmänä, jolloin merkki saadaan vilkkumaan. Kun näyttö on tekstitilassa, Applen videoyksikkö katkaisee väritahdistussignaalin ja tuottaa selvemmän mustavalkoisen näytön.*

Ensimmäisen tekstisivun käyttämä muistialue alkaa osoitteesta 1024 ja ulottuu osoitteeseen 2047. toinen sivu ulottuu osoitteesta 2048 osoitteeseen 3071. Konekielessä(vastaavat heksadesimaaliosoitteet) ensimmäinen sivu ulottuu osoitteesta $400 osoitteeseen $7FF, toinen $800:sta $BFF:aan. Nämä sivut ovat 1024 tavua pitkiä. Ne jotka eivät pelkää päässälaskua, ovat varmaan huomanneet, että näytössä on vain 960 merkkiä. Jäljelle jäävää 64 tavua käytetään väliaikaisesti Applen oheisliitäntäkorttien PROM-muistissa olevien ohjelmien tuottamien tietojen väliaikaisina säilytyspaikkoina.

Valokuvassa 6 näkyvät Applen käyttämät 64 merkkiä.



Valokuva 6. Applen merkkivalikoima

Taulukko 7 antaa merkkien desimaali- ja heksadesimaalikoodit normaalissa, käännetyssä ja vilkkuvassa tulostuksessa.

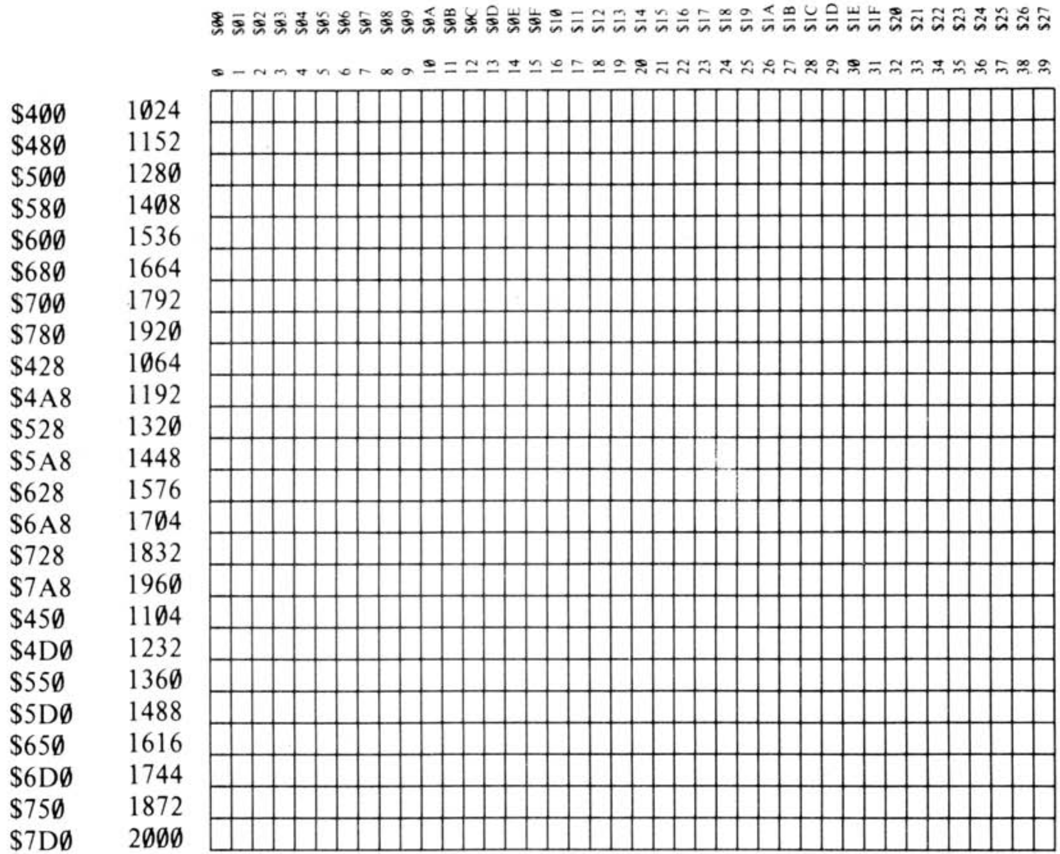
---

* Tätä mahdollisuutta ei ole version Ø levyissä.

Taulukko 7: ASCII-merkit

	Käänteinen				Vilkkuva tekstitila				Normaali tekstitila (Kontrollimerkki)						(pienet kirjaimet)	
Decimal	Ø	16	32	48	64	8Ø	96	112	128	144	16Ø	176	192	2Ø8	224	24Ø
Hex	$ØØ	$1Ø	$2Ø	$3Ø	$4Ø	$5Ø	$6Ø	$7Ø	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
Ø $Ø	@	P		Ø	@	P		Ø	@	P		Ø	@	P		Ø
1 $1	A	Q	!	1	A	Q	!	1	A	Q	!	1	A	Q	!	1
2 $2	B	R	"	2	B	R	"	2	B	R	"	2	B	R	"	2
3 $3	C	S	#	3	C	S	#	3	C	S	#	3	C	S	#	3
4 $4	D	T	$	4	D	T	$	4	D	T	$	4	D	T	$	4
5 $5	E	U	%	5	E	U	%	5	E	U	%	5	E	U	%	5
6 $6	F	V	&	6	F	V	&	6	F	V	&	6	F	V	&	6
7 $7	G	W	'	7	G	W	'	7	G	W	'	7	G	W	'	7
8 $8	H	X	(	8	H	X	(	8	H	X	(	8	H	X	(	8
9 $9	I	Y	)	9	I	Y	)	9	I	Y	)	9	I	Y	)	9
1Ø $A	J	Z	•	:	J	Z	•	:	J	Z	•	:	J	Z	•	:
11 $B	K	[	+	;	K	[	+	;	K	[	+	;	K	[	+	;
12 $C	L	\	,	<	L	\	,	<	L	\	,	<	L	\	,	<
13 $D	M	]	—	=	M	]	—	=	M	]	—	=	M	]	—	=
14 $E	N	^	.	>	N	^	.	>	N	^	.	>	N	^	.	>
15 $F	O	_	/	?	O	_	/	?	O	_	/	?	O	_	/	?

Taulukko 7: ASCII-merkkivalikoima



Kuva 1. Tekstinäytön kartta

Kuva 1 on kartta näytöstä tekstitilassa. Siitä käy ilmi jokaisen näytössä olevan merkin sijaintia vastaavan muistipaikan osoite.

PERUSGRAFIIKKA (LO-RES) TILA

Perusgrafiikassa Apple esittää samat 1024 muistipaikkaa kuin tekstitilassa, mutta eri muodossa. Tässä tilassa jokainen muistin tavu esitetään, ei ASCII-merkkinä vaan kahtena värillisenä ruutuna, jotka sijaitsevat päällekkäin. Näyttö voi esittää 40 ruutua leveän ja 48 ruutua korkean ruutumatriisin. Jokaisen ruudun väri voidaan valita 16:sta vaihtoehdosta. Mustavalkoisessa monitorissa värit näkyvät valkoisina ja harmaan erisävyisinä pisteinä.

Koska jokainen tavu esitetään kahtena ruutuna, on se jaettu kahteen yhtä suureen osaan, joita kutsutaan »nybbleksi». Jokaisen »nybblen» arvo voi vaihdella nollasta viiteentoista. Tavun ensimmäinen »nybble» määrää näytön ylemmän ruudun värin ja jälkimmäinen vastaavasti alemman. Värit on numeroitu nollasta viiteentoista seuraavasti:

Taulukko 8: Perusgrafiikan värit

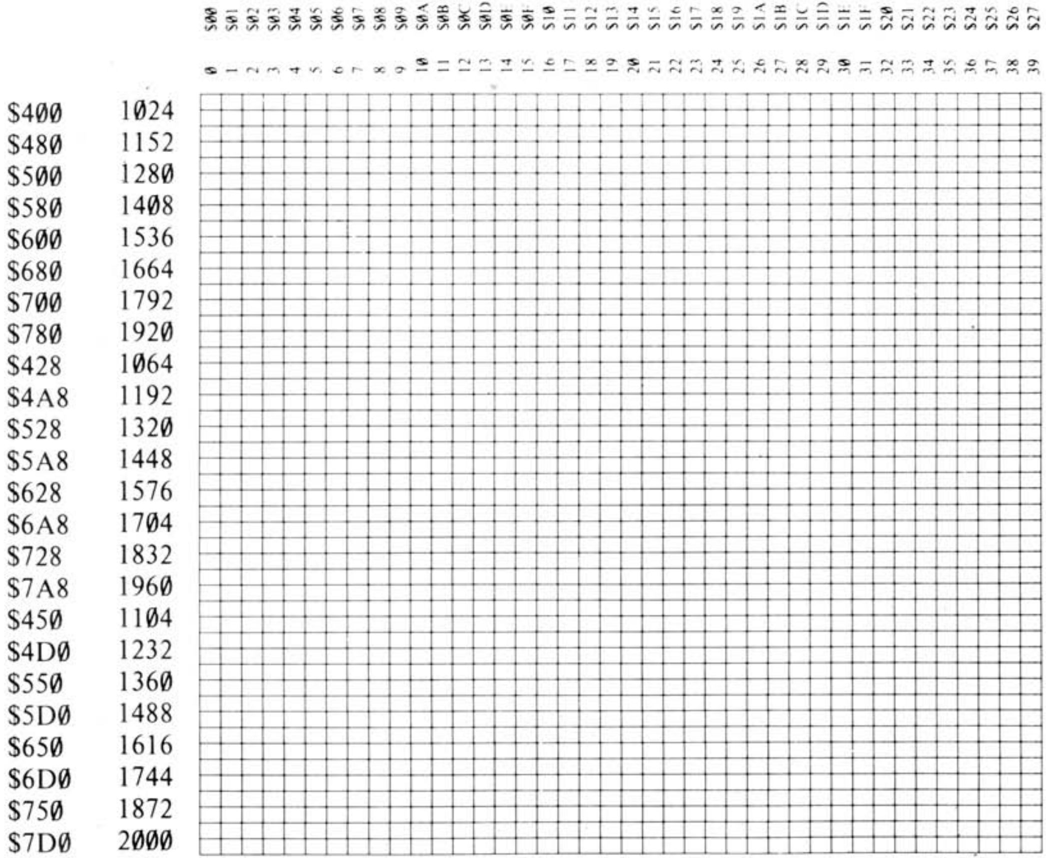
Desim.	Hex	Väri	Desim.	Hex	Väri
Ø	$Ø	Musta	8	$8	Ruskea
1	$1	Sinipunainen	9	$9	Oranssi
2	$2	Tumman sininen	1Ø	$A	Harmaa 2
3	$3	Purppura	11	$B	Vaalean punainen
4	$4	Tumman vihreä	12	$C	Vaalean vihreä
5	$5	Harmaa 1	13	$D	Keltainen
6	$6	Sininen	14	$E	Akvamariini
7	$7	Vaaleansininen	15	$F	Valkoinen

(Värit voivat vaihdella televisiokohtaisesti ennen kaikkea värisävysäätöjen takia. Värejä voidaan säätää emolevyn oikeassa laidassa olevan värien hienosäädön avulla.)

Siis tavu, joka sisältää heksadesimaaliluvun $D8, ilmestyy näyttöön ruskeana ruutuna, jonka alapuolella on keltainen ruutu. Desimaaliaritmetiikkaa käyttäen alemman ruudun värin numero on tavun arvo jaettuna 16:lla. Ylemmän ruudun värin numero on jakojäännös.

Kuva 2 on kartta Applen näytöstä perusgrafiikkatilassa. Kuvasta ilmenee jokaista näytön ruutua vastaavan muistipaikan osoite.

Koska perusgrafiikkka ja teksti käyttävät samaa muistialuetta, aiheuttaa vaihtaminen tilasta toiseen mielenkiintoisia seurauksia. Jos esim. näyttö on perusgrafiikassa täynnä värillisiä ruutuja ja TEKSTI/GRAFIIKKA -kytkin muutetaan tekstitilaan, tulostuu näyttö täyteen satunnaisilta vaikuttavia numeroita, joista jotkut ovat vilkkuvia tai väriltään käänteisiä. Vastaavasti tekstiä täynnä oleva näyttö kuvautuu perusgrafiikassa pitkinä vaakasuorina harmaina, vaaleanpunaisina, vihreinä tai keltaisina juovina. sisältäen satunnaisia, mielivaltaisen värisiä ruutuja siellä täällä.



Kuva 2. Perusgrafiikkanäytön kartta

TARKKUUSGRAFIIKKA

Applella on toisenkintyyppinen grafiikka, jota kutsutaan tarkkuusgrafiikaksi (joskus Hi-res:ksi). Kun Apple on tarkkuusgrafiikkatilassa, voidaan näytössä esittää 53760 pistettä, jotka muodostavat 280 pistettä leveän ja 192 pistettä korkean matriisin. Pisteet voivat olla mustia, valkoisia, violetteja, vihreitä, punaisia tai sinisiä, joskin yksittäisten pisteiden värimahdollisuudet ovat rajoitetut.

Tarkkuusgrafiikka ottaa käyttämänsä tiedon 8192 tavua laajalta muistialueelta, jota kutsutaan kuvapuskuriksi. Kuvapuskureita on kaksi kappaletta: toinen sivua 1 ja toinen sivua 2 varten. Molemmat puskurit ovat itsenäisiä ja erillisiä tekstin ja perusgrafiikan käyttämistä muistialueista. Ensimmäisen sivun kuvapuskuri alkaa muistipaikasta 8192 ja päättyy muistipaikkaan 16383. Toisen sivun puskuri seuraa välittömästi tämän perässä ja ulottuu muistipaikasta 16384 muistipaikkaan 24575. Niille, joilla sattuu olemaan kuusitoista sormea: ensimmäinen sivu täyttää muistipaikat $2000–$3FFF ja toinen muistipaikat $4000–$5FFF. Jos Applessa on muistitilaa 16K (16.384 tavua) tai vähemmän, ei toinen sivu ole käytettävissä. Jos muistitilaa on alle 16 K, puuttuu koko tarkkuusgrafiikka.

Jokainen näytön piste edustaa yhtä kuvapuskurin bittiä. Seitsemän kahdeksasta bitistä joka tavussa esitetään näytössä ja jäljellejäävää bittiä käytetään esitettyjen pisteiden värin määräämiseen. Näytön jokaisella rivillä esitetään neljäkymmentä tavua. Tavun vähiten merkitsevä (ensimmäinen) bitti sijoitetaan näytön vasempaan reunaan, sen jälkeen toinen bitti jne. Eniten merkitsevää (kahdeksatta) bittiä ei esitetä. Tämän jälkeen tulevan seuraavan tavun ensimmäinen bitti sijoitetaan näytön vasempaan reunaan, sen jälkeen toinen bitti jne. Eniten merkitsevää (kahdeksatta) bittiä ei esitetä. Tämän jälkeen tulee seuraavan tavun ensimmäinen bitti jne. Näytön jokaisella 192 rivillä esitetään kullakin 280 pistettä.

Mustavalkoisessa monitorissa pisteet, joita vastaava bitti on »päällä» (eli = 1) näkyvät valkoisina, ne jotka ovat »poissa päältä» (eli = 0) näkyvät mustina. Värimonitorilla tai -televisiolla asia ei ole aivan yhtä yksinkertainen. Jos bitti on »poissa päältä», sitä vastaava piste on aina musta. Jos bitti kuitenkin on »päällä», sen väri riippuu pisteen sijainnista näyttöruudulla. Jos piste on vasemmanpuoleisimmassa sarakkeessa (jota kutsutaan sarakkeeksi 0) tai missä tahansa parillisessa sarakkeessa, se on violetti. Jos se on oikeanpuoleisimmassa sarakkeessa (sarakkeesa 279) tai missä tahansa parittomassa sarakkeessa, se on vihreä. Jos kaksi pistettä on rinnakkain, ne ovat molemmat valkoisia. Jos bitti, jota ei esitetä näytössä, on »päällä», korvautuvat violetti ja vihreä sinisellä ja punaisella*. Tarkkuusgrafiikalla on siis käytössään kuusi väriä seuraavin rajoituksin:

1) Parillisessa sarakkeessa olevat pisteet ovat mustia, violetteja tai sinisiä.

2) Parittomassa sarakkeessa olevat pisteet ovat mustia, vihreitä tai punaisia.

3) Jokainen tavu on violetti/vihreä tai punainen/sininen. Vihreää ja sinistä, vihreää ja punaista, violettia ja sinistä tai violettia ja punaista ei voi sekoittaa samaan tavuun.

---

* Versio Ø -levyissä ei ole sinistä ja punaista ja näin ollen kahdeksannen bitin arvolla ei ole merkitystä.

4) Kaksi vierekkäistä värillistä pistettä ovat aina valkoisia, vaikka ne kuuluisivat eri tavuihin.

5) Applen eurooppalaisissa versioissa nämä säännöt pätevät muuten,mutta värit voivat olla toiset.

Kuva 3 esittää Applen näyttöä tarkkuusgrafiikkatilassa sekä kutakin tavua vastaavien muistipaikkojen osoitteet.

MUITA SYÖTTÖ/TULOSTUSMUOTOJA

Applen sisäänmenot ja ulostulot	
Sisäänmenot:	Kasettisisäänmeno
	Kolme yhden bitin digitaalisisäänmenoa
	Neljä analogiasisäänmenoa
Ulostulot:	Kasettiulostulo
	Sisäänrakennettu kaiutin
	Neljä Annunciator-ulostuloa
	Utility Strobe -ulostulo

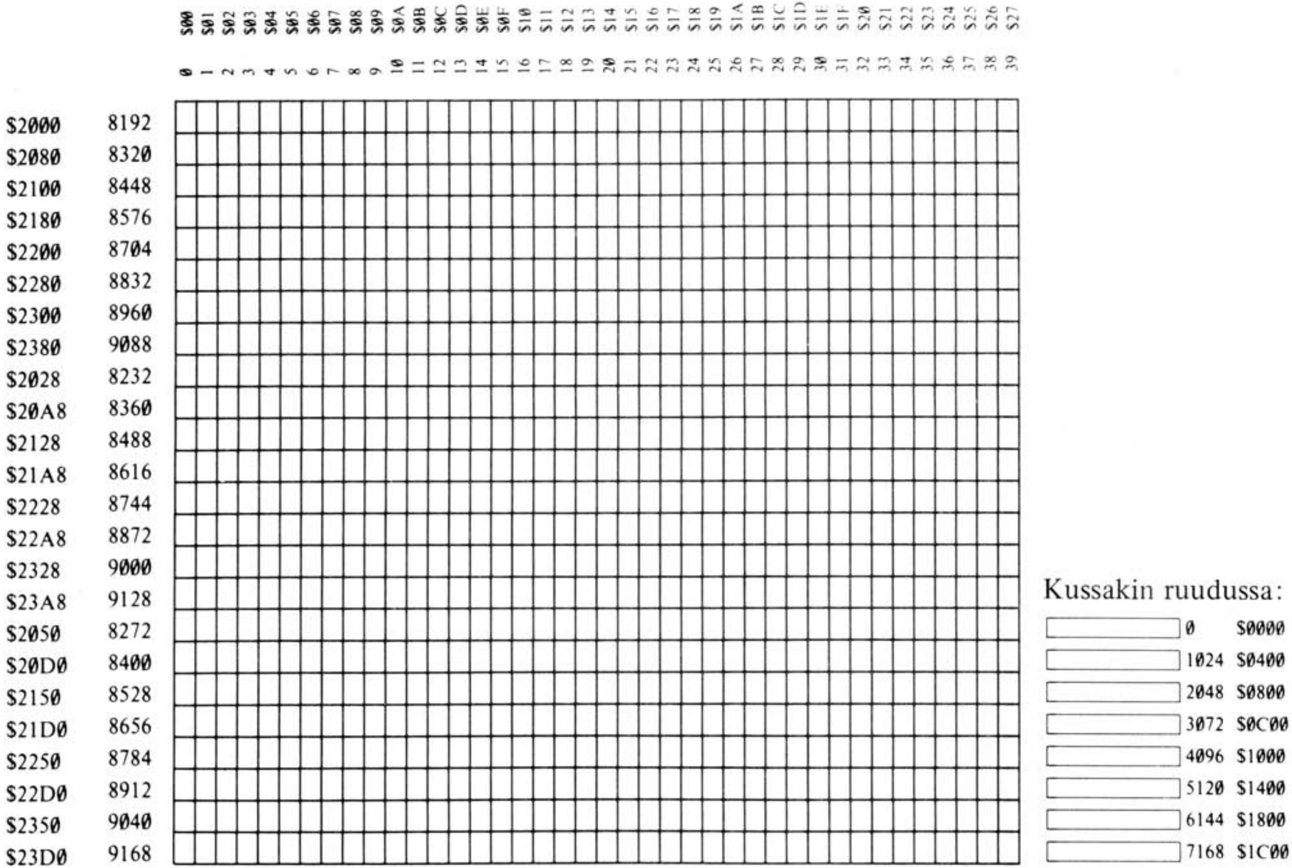
KAIUTIN

Applen sisällä, vasemmassa laidassa näppäimistön alla on pieni 8 ohmin kaiutin. Se on yhdistetty Applen sisäiseen elektroniikkaan niin että ohjelma voi käyttää sitä erilaisten äänien tuottamiseen.

Kaiutinta ohjataan »pehmeällä» kytkimellä. Kytkin voi asettaa kaiuttimen paperikartion kahteen asentoon: »sisään» ja »ulos». Tämä kytkin ei ole samanlainen kuin ne »pehmeät» kytkimet, jotka valvovat eri näyttötiloja vaan ns. vaihtokytkin. Joka kerta kun ohjelma viittaa tähän kytkimeen liittyvän muistipaikan osoitteeseen, kaiuttimen tila muuttuu: jos se oli »sisään» siitä tulee »ulos» ja kääntäen. Joka kerta kun tila muuttuu, kaiutin tuottaa pienen naksahduksen. Viittaamalla tähän osoitteeseen jatkuvasti ja taajaan, voidaan tuottaa tasainen ääni.

Kaiuttimen pehmeä kytkin on yhdistetty muistipaikkaan 49200. Mikä tähansa viittaus tähän osoitteeseen (tai yhtäpitäviin osoitteisiin — 16336 tai heksadesimaali $C030) saa kaiuttimen naksahtamaan.

Ohjelma viittaa osoitteeseen luku- tai kirjoituskäskyllä. Luettu tai kirjoitettu tieto on merkityksetöntä, sillä pelkkä osoite kääntää kytkimen. On syytä huomata, että kirjoitusoperaatiossa Applen 6502 mikroprosessori itse asiassa lukee muistipaikan sisällön ennen kirjoittamistaan, joten jos kytkintä käännetään kirjoituskäskyllä, sitä käännetäänkin kahdesti. Tämä merkitsee sitä, että vaihtotyyppisille kytkimille — kuten kaiuttimen kytkin — kirjoittaminen sitä vastaavaan muistipaikkaan jättää kytkimen samaan asentoon kuin missä se oli ennen kirjoittamista.



Kuva 3. Tarkkuusgrafiikkanäytön kartta

KASETTILIITÄNTÄ

Applen emolevyn takalaidassa videoliittimen oikealla puolella on kaksi pientä mustaa liitintä, joissa on tekstit IN ja OUT. Nämä ovat naaraspuolisia pienoisjakkiliittimiä, joihin voidaan kiinnittää johto, jonka kummassakin päässä on vastaavat urosliittimet. Johdon toinen pää voidaan liittää tavalliseen kasettisoittimeen. Näin Apple voi taltioida tietoa kasetille ja myöhemmin lukea sitä.

OUT-liitin on liitetty ohjelmalliseen (»pehmeään») kytkimeen. Tämä on vaihtokytkin kuten kaiutinkytkinkin (vrt. kohta KAIUTIN). Kasetin ulostulo-kytkin voidaan kääntää viittaamalla muistipaikkaan numero 49184 (vastaavasti -16352 tai hexadesimaalisena $C020). Tähän osoitteeseen viittaaminen saa OUT-liittimen jännitteen kasvamaan nollasta 25 millivolttiin (= 1/40 volttiin) tai laskemaan 25 millivoltista takaisin nollaan. Jos johdon toinen pää on kasettinauhurin mikrofoniliitännässä ja nauhuri on kytketty äänittämään, taltioituu nauhalle pieni naksahdus. Viittaamalla osoitteeseen jatkuvasti ja tiuhaan voi ohjelma tuottaa jatkuvan äänen nauhalle. Vaihtelemalla äänen korkeutta ja kestoa voidaan nauhalle siirtää tietoa ja säilyttää se myöhempää käyttöä varten. System Monitorissa on erillinen aliohjelma tätä varten. Ohjelmasta on kerrottu tarkemmin sivulla 46.

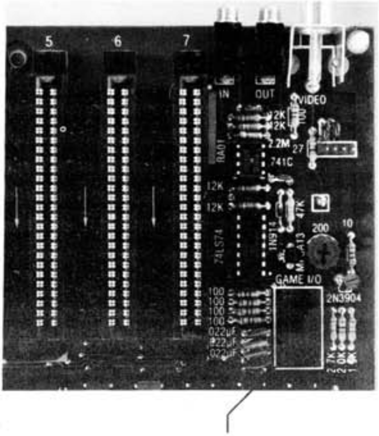
On syytä muistaa, että jos kasettiulostulon (OUT) ohjelmallista kytkintä käännetään kirjoittamalla asianomaiseen muistipaikkaan, tulee nauhoitukseen itse asiassa kaksi naksahdusta. Syy on selitetty kaiuttimen yhteydessä edellä. Tätä kytkintä käännettäessä tulisi käyttää pelkästään lukuoperaatioita.

Toista liitintä, jossa on merkintä IN, voidaan käyttää nauhoituksen »kuuntelemiseen». Se antaa mahdollisuuden kuunnella äänet nauhalta, muuttaa ne tiedoiksi ja varastoida tiedot muistiin. Näin kasetille varastoitu ohjelma tai informaatio voidaan lukea ja käyttää uudelleen.

Sisääntulopiirit ottavat 1 voltin signaalin nauhurin kuulokeliittimestä ja muuttavat sen ykkösiä ja nollia sisältäväksi jonoksi. Joka kerta kun signaali muuttuu positiivisesta negatiiviseksi tai kääntäen, piirin tila muuttuu: jos se syötti ykkösiä, se alkaa syöttää nollia ja kääntäen. Ohjelma voi tarkastaa piirin tilan tutkimalla muistiapikan numero 49248 (vastaavasti 16288 tai heksadesimaalisena $C060) sisällön. Jos arvo on 128 tai suurempi, tila on »yksi», jos taas pienempi, se on »nolla». Vaikka BASIC-ohjelmat voivatkin tarkastaa ko. tilan, ne ovat tavallisesti liian hitaita saadakseen mitään selvyyttä lukemastaan. System Monitorissa on kuitenkin ohjelma, joka lukee tietoja nauhalta ja tulkitsee ne. Tätä ohjelmaa kuvaillaan tarkemmin sivulla 47.

PELIEN I/O-LIITIN

Pelien I/O-liittimen avulla voidaan syötössä ja tulostuksessa käyttää erikoisvälineitä. Tämä parantaa ohjelmien tehokkuutta yleensä ja peliohjelmien tehoa erityisesti. Liitin antaa myös mahdollisuuden käyttää kolmea yhden bitin sisäänmenoa, neljää yhden bitin ulostuloa, data strobea ja neljää analogisisäänmenoa. Kaikkia näitä toimintoja voidaan ohjata ohjelmallisesti. Applen mukana toimitetaan kaksi peliohjainta, jotka kytketään johdollaan pelien I/O-liittimeen. Ohjaimien säätimet on yhdistetty liittimen analogiasisäänmenoihin, painonapit taas on kytketty yhden bitin sisäänmenoihin.



Valokuva 7. Pelien I/O-liitin

KUTSU-ULOSTULOT

Neljää yhden bitin ulostuloa sanotaan kutsu-ulostuloksi. Kukin kutsu-ulostulo voidaan liittää jonkin elektronisen laitteen sisäänmenoon, tai sitä voidaan käyttää ohjaamaan lamppuja, releitä, kaiuttimia jne.

Jokaista näistä ulostuloista ohjataan ohjelmallisella kytkimellä. Kytkinten osoitteet on järjestetty neljäksi pariksi, yksi pari kutakin kytkintä kohti. Jos parin ensimmäiseen osoitteeseen viitataan, vastaavan ulostulon tila käännetään »pois», jos taas viitataan toiseen osoitteeseen, tila käännetään »päälle». Kun kutsu-ulostulo on »pois», pelien I/O-liittimen vastaavan nastan jännite on likimain 0. Kun ulostulo taas on »päällä», jännite on noin 5 volttia. Kutsu-ulostulossa vallitsevaa tilaa ei voi selvittää millään yksinkertaisella keinolla. Kutsu-ulostulon ohjelmalliset kytkimet ovat:

Taulukko 9: Kutsu-ulostulojen osoitteet				
	Tila	Osoite: Desim.		Hex
Ø	ei	4924Ø	-16296	$CØ58
	on	49241	-16295	$CØ59
1	ei	49242	-16294	$CØ5A
	on	49243	-16293	$CØ5B
2	ei	49244	-16292	$CØ5C
	on	49245	-16291	$CØ5D
3	ei	49246	-1629Ø	$CØ5E
	on	49247	-16289	$CØ5F

YHDEN BITIN SISÄÄNMENOT

Kolmesta yhden bitin sisäänmenosta jokainen voidaan liittää johonkin elektroniseen laitteeseen tai painonappiin. Mikä tahansa näistä sisäänmenoista voidaan lukea konekielellä tai BASIC-ohjelmalla samalla tavoin kuin kasetilta (käsitelty edellä). Sisäänmenoja vastaavien muistipaikkojen osoitteet ovat 49249:stä 49251:een (-16287:stä -16285:een tai heksadesimaalisena $C061:stä $C063:een).

ANALOGIA-SISÄÄNMENOT

Nelja analogiasisäänmenoa voidaan liittää 150K ohmin säädettävään vastukseen tai potentiometriin. Ajastimena käytetty yksiasentoinen multivibraattori käyttää hyväkseen resistanssin vaihtelua +5 voltin jännitteen ja kyseisen sisäänmenon välillä. Kun vastus sisäänmenossa muuttuu, se muuttaa vastaavan ajastinpiirin toimintaa. Konekieliset ohjelmat pystyvät havaitsemaan nämä aikavakioiden muutokset ja johtamaan siitä potentiometrin asentoa vastaavat numeeriset arvot.

Ennen kuin ohjelma voi aloittaa potentiometrin asennon lukemisen, sen täytyy nollata ajastinpiirit. Muistipaikka numero 49264 (-16272 tai heksadesimaali $C070) tekee juuri tämän. Kun ajastinpiirit nollataan, muuttuvat muistipaikkojen 49252–49255 (-16284:stä -16281:een tai $C064–$C067) arvot suuremmiksi kuin 128 (niiden korkein bitti asetetaan). 3.060 millisekunnissa näiden neljän muistipaikan arvojen pitäisi pudota alle 128:n. Tähän kuluva täsmällinen aika riippuu muistipaikkaan liitetyn peliohjaimen asennosta. Jos analogiasisäänmenoon kytketyn potentiometrin vastusarvo on suurempi kuin 150 K-ohmia, tai mitään potentiometriä ei olekaan, peliohjaimia vastaavien muistipaikkojen arvot saattavat jäädä pysyvästi nollaa suuremmiksi.

TAHDISTUS-ULOSTULO (STROBE)

On olemassa vielä yksi ulostulo, jota kutsutaan C∅40STROBE:ksi. Se tila on tavallisesti 5 volttia, mutta voi pudota nollaan puolen mikrosekunnin ajaksi konekielisen tai BASIC-ohjelman valvonnassa. Tämä tahdistin voidaan laukaista viittaamalla muistipaikkaan numero 49216 (-16320 tai $C04E). On syytä muistaa, että tähän muistipaikkaan suoritettu kirjoitusoperaatio laukaisee tahdistuksen kahdesti (tästä ilmiöstä on tarkempi kuvaus kaiutinta koskevassa luvussa).

Taulukko 10: I/O:n erikoismuistipaikat

Toiminta	Osoite: Desim.		Hexa	Luku / Kirjoitus
Kaiutin	49200	-16336	$C030	L
Kasetti-sisäänmeno	49184	-16352	$C020	L
Kasetti-ulostulo	49256	-16288	$C060	L
Kutsu-ulostulot*	49240	-16296	$C058	L/K
	...	...	...	
	49247	-16289	$C05F	
Lippu-sisäänmenot	49249	-16287	$C061	L
	49250	-16286	$C062	L
	49251	-16285	$C063	L
Analogia-sisäänmenot	49252	-16284	$C064	L
	49253	-16283	$C065	
	49254	-16282	$C066	
	49255	-16281	$C067	
Analogia-nollaus	49264	-16272	$C070	L/K
Utility Strobe	49216	-16320	$C040	L

APPLEN VERSIOT

Apple II:sta on muutamia muunnelmia. Jotkut niistä eroavat itse tietokoneen osalta, toiset taas ovat erilaisia ohjelmistoltaan. Perusversiot ovat seuraavat:

AUTOSTART ROM / MONITOR ROM

Kaikissa Apple II Plus -järjestelmissä on Autostart Monitor ROM. Sen sijaan joissakin muissa Apple järjestelmissä on tämän asemesta Apple System Monitor ROM. Viimemainitusta versiosta puuttuu muutamia Autostart ROMin piirteitä, mutta toisaalta siinä on ominaisuuksia, joita Autostart ROMissa ei ole. Tärkeimmät erot näiden välillä on lueteltu seuraavalla sivulla.

* ks. edellinen taulukko

- **Editointikontrollit** ESC-I, J, K ja M, jotka siirtävät kursoria ylös, vasemmalle, oikealle ja alas, puuttuvat vanhasta Monitor ROMista.

- **Tulostuksen keskeytys.** Tulostuksen keskeytys -rutiini (suoritetaan CTRL S:llä), joka mahdollistaa tauon useimmissa BASIC- tai konekielisissä ohjelmissa ja listauksissa, puuttuu vanhasta Monitor ROMista.

- **Nollaus (Reset) -rutiini.** Kun Applen virta kytketään päälle tai painetaan RESET näppäintä, vanha Monitor ROM siirtyy suoraan Apple System Monitoriin sen sijaan, että suorittaisi »lämpimän» tai kylmän» käynnistyksen, joita kuvaillaan kappaleessa NOLLAUS-RUTIINI sivulla 36.

Vanha Monitor ROM sisältää kuitenkin STEP ja TRACE -vianetsintärutiinit, joita käsitellään sivulla 51. Autostart ROM ei hyväksy näitä käskyjä.

VERSIO Ø / VERSIO 1 -LEVY

Version 0 Apple II -levystä puuttuu muutamia piirteitä, jotka ovat uudessa 1-versiossa. Jos haluat tietää, kumpaa versiota juuri Sinun Applesi emolevy on, avaa Applen kansi ja katso levyn oikeaa yläkulmaa. Vertaa sitä sivulla 10 olevaan kuvaan 4. Jos siinä ei ole metallista videoliittimen juotostornia nelinapaisen videoliittimen ja videosäädin-potentiometrin välissä, kyseessä on version 0 Apple.

Tärkeimmät erot Version 0 ja Version 1 -levyjen välillä ovat seuraavat:

- **Värisalpa.** Kun Applen näyttö on tekstitilassa, Version 0 -levy jättää väritahdistussignaalin päälle. Seurauksena on kirjaimien epämääräinen väri tai niiden värilliset reunukset.

- **Käynnistyksen nollausrutiini.** Version 0 -levyssä ei ole yksikköä, joka automaattisesti käynnistäisi nollausrutiinin, kun virta kytketään päälle. Sen sijaan sinun täytyy painaa RESETiä kerran, ennen kuin alat käyttää Appleasi.

  Lisäksi, kun Version 0 Apple käynnistetään, näppäimistö tulee aktiiviseksi, aivan kuin satunnainen merkki olisi kirjoitettu. Kun Apple alkaa etsiä syöttötietoa näppäimistöltä, se hyväksyy tämän merkin kuin se olisi kirjoitettu. Sen poistamiseksi on syytä painaa CTRL X:ää aina RESETin jälkeen, kun virta pannaan päälle.

- **Tarkkuusgrafiikan värit.** Version 0 -levyllä varustetut Applet voivat aikaansaada tarkkuusgrafiikassa vain neljä väriä: musta, valkoinen, vihreä ja violetti. Jokaisen näytetyn tavun korkein bitti (vertaa sivu 19) jätetään huomioimatta.

- **24K »muistiongelma».** Järjestelmissä, joissa on Version 0 -levy sekä joissa on käytössä RAM-muistia 20K tai 24K, BASIC luulee muistia kuitenkin olevan enemmän. Tarkempi kuvaus tästä ongelmasta löytyy kappaleesta »Muistin organisointi» sivulta 72.

- **50 Hz Applet.** Version 0 -levyssä ei ole foliosiltoja ja kytkentäjohtoja, jotka voitaisiin katkaista ja juottaa niin, että VIDEO ulostulosignaali vastaisi eurooplaisia PAL/SECAM -standardeja. Siitä puuttuu myös kolmas videoliitin, metallinen juotostorni nelinapaisen videoliittimen edessä.

- **Kaiutin- ja kasettiliitäntä.** Version 0 -levyllä varustetuille Appleille jokainen kaiuttimen aikaansaama ääni on samalla signaali kasettinauhurin OUT-liittimelle. Jos nauhurissa on äänitys päällä, mikä tahansa kaiuttimen aikaansaama ääni siirtyy myös nauhalle.

- **Kasettisisäänmeno.** Kasettiliitännän sisäänmenopiiri on muutettu siten, että se reagoi tarkemmin heikkoihin signaaleihin.

EROT VIRTALÄHTEISSÄ

Joissakin Appleissa on virtalähteestä versio, joka hyväksyy vain 110 voltin verkkojännitteen. Niissä ei ole jännitteen valitsinta virtalähteen takasivussa.

APPLE II PLUS

Apple II Plus on normaali Apple II tietokone, jossa on Version 1 -levy, Autostart Monitor ROM ja kielenä Applesoft II BASIC Apple Integer BASICin asemesta. Apple II Plus -tietokoneen eurooppalaiset mallit on varustettu 110/220 voltin virtalähteellä. Apple Mini -Assembler, liukulukuesitys ja SWEET-16 tulkki, jotka ovat Integer BASIC ROMissa, eivät kuulu Apple II Plus -tietokoneen perusversioon.

LUKU 2
KESKUSTELU APPLEN KANSSA

Melkeinpä jokainen Applen ohjelma ja kieli tarvitsee jonkinlaista syöttötietoa näppäimistöltä ja jonkin keinon tulostaa informaatiota näytölle. Applen ROM-muistiin on taltioitu joukko ohjelmia, jotka hoitavat suurimman osan tavanomaisesta syöttö- ja tulostusrutiineista.

Näitä alirutiineita kutsutaan eri nimillä. Nimet ovat ohjelmien tekijöiden antamia siihen aikaan kun rutiinit laadittiin. Apple itse ei muista eikä tunne näitä nimiä, mutta meidän kannaltamme on kätevää käyttää alirutiineista niiden omia nimiä.

STANDARDI-ANTO (OUTPUT)

Tavanomaisesta tulostuksesta huolehtiva alirutiini on nimeltään COUT. COUT näyttää isot kirjaimet, numerot, ja symbolit näytössä joko normaalissa tai käänteisessä tilassa. Se ei tulosta kontrollimerkkejä, poikkeuksena RETURN, äänimerkki, rivinsiirto ja peruutusmerkki.

COUTilla on oma näkymätön tulostuskursorinsa* (paikka, johon seuraava merkki sijoitetaan). Joka kerta kun COUTia kutsutaan, se sijoittaa yhden merkin näyttöön kursorin kulloiseenkin sijaintipisteeseen — ja peittää merkin, joka siinä alunperin oli — ja siirtää kursoria yhden askeleen oikealle. Jos kursori siirtyy yli näytön oikean laidan, COUT siirtää sen seuraavan rivin alkuun. Jos kursori ohittaa näytön alalaidan, näyttöä »rullataan» rivin verran ylöspäin ja kursori sijoitetaan uuden, tyhjän alarivin ensimmäisen merkin kohdalle.

Kun COUT vastaanottaa RETURN-merkin, se siirtää kursorin seuraavan rivin alkuun. Jos kursori putoaa alareunan alapuolelle, näyttöä rullataan, kuten edellä on kuvattu.

TULOSTUKSEN PYSÄYTYS (STOP-LIST)

Kun joku ohjelma tai kieli lähettää RETURNin COUTille, COUT lukee nopeasti näppäimistön. Jos edellisen lukuoperaation jälkeen on kirjoitettu [CTRL S], COUT pysähtyy ja odottaa uuden näppäimen painamista. Tätä kutsutaan Stop-list** -rutiiniksi. Kun seuraavaa merkkiä painetaan, COUT tulostaa RETURNin ja jatkaa normaalisti. Se merkki, jolla Stop-list tila katkaistiin, on merkityksetön, elllei se ole [CTRL C]. Jos näin on, COUT siirtää merkin tulostusta lähettävälle ohjelmalle tai kielelle. Tämä antaa mahdollisuuden päättää BASIC-ohjelma tai listaus pysäytyksen jälkeen.

Rivinsiirto saa COUTin siirtämään kursoria rivin alaspäin ilman vaakasuoraa liikettä. Kuten aina, alareunan ohittaminen saa näytön rullaamaan ja kursorin siirtymään uuden rivin alkuun.

Peruutusmerkki siirtää kuviteltua kursoria yhden askeleen vasemmalle. Jos kursori ylittää vasemman laidan, se asetetaan edellisen rivin oikeanpuoleisimpaan pisteeseen. Jos edellistä riviä ei ole (jos kursori oli ylärivillä), näyttö ei rullaa alaspäin vaan kursori asetetaan uudelleen ylimmän rivin oikeanpuoleisimpaan pisteeseen.

* Latinasta cursus = »juoksija»

** Srop-list -piirre puuttuu Appleista, joissa ei ole Autostart ROMia.

Kun COUTille lähetetään äänimerkki (CTRL G), se ei kajoa näyttöön lainkaan vaan tuottaa äänen kaiuttimella. Äänen taajuus on1000HZ ja se kestää 1/10 sekuntia. Tulostuskursori ei tällöin liiku.

TEKSTI-IKKUNA

Edellä käydyssä tulostuskursorin liikkeitä koskevassa osassa käytetyt sanat »oikea», »vasen», »yläreuna», »alareuna» tarkoittivat tavallisen 40 merkkiä leveän ja 24 merkkiä korkean näytön fyysisiä reunoja. On kuitenkin olemassa keino kertoa COUTille, ettei haluta käyttää koko 960 merkin näyttöä vaan ainoastaan osaa siitä. Tällaista näytön erillistä osaa kutsutaan ikkunaksi. Ohjelma tai kieli voi määritellä ikkunan yläreunan, alareunan, vasemman laidan ja leveyden varastoimalla ao. tiedot neljään muistipaikkaan. Kun tämä on tehty, COUT käyttää uusia arvoja laskiessaan näytön kokoa. Se ei koskaan kirjoita mitään ikkunan ulkopuolelle, ja kun sen on rullattava näyttöä, se rullaa vain ikkunassa olevaa tekstiä. Näin ohjelmat voivat valvova tekstin sijoittelua ja suojella joitakin alueita päällekirjoittamiselta.

Muistipaikka numero 32 (heksadesimaali $20) sisältää ikkunan vasemmanpuoleisimman sarakkeen numeron. Se on normaalisti 0, mikä tarkoittaa näytön vasemmanpuoleisinta saraketta. Se ei koskaan saa olla suurempi kun 39 (heksadesimaali $27). Muistipaikka numero 33 (heksadesimaali $21) sisältää ikkunan leveyden sarakkeissa lausuttuna. Se on normaalisti 40 (heksadesimaali $28). On syytä tarkistaa, ettei ikkunan vasemman reunan ja leveyden summan ylitä neljääkymmentä! Jos näin käy, COUT saattaa sijoittaa merkkejä muistipaikkoihin, jotka eivät esiinny näytössä ja siten sotkea muistissa sijaitsevan ohjelman, tai tiedostojen tiedot.

Muistipaikka 34 (heksadesimaali $22) sisältää ikkunan ylimmän rivin numeron. Se on normaalisti 0, mikä tarjoittaa näytön ylintä riviä. Muistipaikka 35 (heksadesimaali $23) sisältää näytön alimman rivin numeron (plus yksi), normaalisti siis 24 (heksadesimaali $18). Kun muutat ikkunan kokoa, pidä huoli siitä, että tiedät missä tulostuskursori on ja että se on uuden ikkunan sisällä.

Taulukko 11: Teksti-ikkunan erikoismuistipaikat

Toiminta	Muistipaikka Desim.	Hex	Minimi/Normaali/Maksimiarvo Desim.	Hex
Vasen reuna	32	$20	0/0/39	$0/$0/$17
Leveys	33	$21	0/40/40	$0/$28/$28
Yläreuna	34	$22	0/0/24	$0/$0/$18
Alareuna	35	$23	0/24/24	$0/$18/$18

MUSTAVALKOINEN NÄKEMYS

COUT-alirutiini voi kirjoittaa saamansa merkit joko normaalissa tai käänteisessä tekstitilassa (ks. sivu 14). Tulostuksen muoto määrytyy muistipaikan numero 50 (heksadesimaali $32) sisällön perusteella. Jos sen arvo on 255 (heksadesimaali $FF), COUT käyttää normaalitilaa. Jos se on 63, COUT käyttää käänteistä tilaa. On syytä huomata, että tämän muistipaikan muutos vaikuttaa vain niihin merkkeihin, jotka kirjoitetaan muutoksen jälkeen. Jos muistipaikkaan numero 50 sijoitetaan muita arvoja, seuraukset ovat eriskummalisia: arvo 127 saa kirjaimet vilkkumaan ja muut merkit tulostumaan käänteisinä; mikä tahansa muu arvo saa COUTin hylkäämään kaikki tai osan normaalista merkkivalikoimastaan.

Taulukko 12: Normaali/käänteinen-tilan ohjausarvot

Arvo: Desim.	Hex	Vaikutus:
255	$FF	COUT tulostaa merkit normaalitilassa.
63	$3F	COUT tulostaa merkit käänteisessä tekstitilassa.
127	$7F	COUT tulostaa kirjaimet vilkkuvina, muut merkit käänteisinä.

Normaali/käänteinen maski — kuten sitä kutsutaan — toimii lisäämällä loogisen JA-operaation muistipaikassa 50 olevien bittien ja tulostuvan merkin bittien väliin. Jokainen muistipaikassa 50 oleva bitti, joka on looginen nolla, muuttaa myös vastaavan bitin tulostuvan merkin koodissa nollaksi riippumatta siitä, mikä se alunperin oli. Jos muistipaikka 50 siis sisältää luvun 63 (heksadesimaali $3F tai binaarinen 00111111), kaksi korkeinta bittiä jokaisen merkin koodissa »sammutetaan». Tämän seurauksena näyttöön tulevien merkkien koodit ovat kaikki 0:n ja 63:n välillä. Kuten ASCII-merkkien koodit sisältävästä taulusta (taulu 7 sivulla 15) näkyy, kaikki nämä merkit ovat käänteisessä tilassa.

STANDARDI-OTTO (INPUT)

On itse asiassa kaksi alirutiinia, jotka hakevat standardimuotoista ottoa: RDKEY, joka hakee yksittäisen näppäimen painalluksen ja GETLN, joka kasaa joukon näppäimenpainalluksia informaatiokimpuksi, jota kutsutaan syöttöriviksi.

RDKEY

RDKEY-alirutiinin ensisijainen tehtävä on odottaa käyttäjän näppäimenpainallusta ja ilmoittaa sitten sitä kutsuneelle ohjelmalle syötetyn merkin koodi. Tämä lisäksi RDKEYllä on myös kaksi muuta tehtävää:

1. *Syötön pyyntö (input prompting).* Kun RDKEY aktivoidaan, se ensimmäiseksi tekee tulostusosoittimen (cursor) näkyväksi. Tällä saavutetaan kaksi päämäärää: se muistuttaa käyttäjälle, että Apple odottaa näppäimenpainallusta ja se sijoittaa myös kaipaamansa oton tiettyyn paikkaan näytöllä. Useimmissa tapauksissa syötön tilaosoitus (prompt) sijaitsee lähellä sanaa tai lausetta, joka ilmoittaa, mitä tietoa ajossa oleva ohjelma tai kieli kaipaa. Syöttö kursori itse on vilkkuva esitys siitä merkistä, joka sillä hetkellä sattui olemaan tulostuskursorin alla. Tavallisesti se on blankko, joten syöttökursori on usein vilkkuva neliö. Kun

käyttäjä painaa näppäintä, RDKEY siirtää tunnollisesti kursoria ja toimittaa painetun näppäimen arvon sitä pyytäneelle ohjelmalle. On syytä muistaa, että tulostuskursori on tietty kohta näytöllä, kun taas syöttökursori on vilkkuva merkki. Ne liikkuvat yleensä rinnan ja ovat harvoin erillään toisistaan, mutta kun syöttökursori katoaa, tulostuskursori on yhä toimiva.

2. *Satunnaislukujen muodostus.* Odottaessaan käyttäjältä näppäimen painallusta RDKEY lisää jatkuvasti ykköstä kahteen muistipaikkaan. Kun näppäintä lopulta painetaan, nämä kaksi muistipaikkaa yhdessä esittävät lukua väliltä 0 -65 535. Luvun tarkkaa arvoa on mahdoton sanoa etukäteen. Monet ohjelmat ja kielet käyttävät tätä lukua satunnaislukugeneraattorin perustana. RDKEYn avulla satunnaistetut muistipaikat ovat numerot 78 ja 79 (heksadesimaali $4E ja $4F).

GETLN

Suurin osa Applen keräämistä ottotiedoista kerätään syöttöriveiksi kutsutuiksi informaatiokimpuiksi. GETLN-niminen Apple ROMin alirutiini hakee syöttörivin näppäimistöltä,ja saatuaan sen palaa sitä kutsuneeseen ohjelmaan. GETLN:ssä on paljon ominaisuuksia ja pikkuhienouksia,ja on hyvä tuntea kaikki sen tarjoamat mahdollisuudet.

Kun GETLNiä kutsutaan, se kirjoittaa ensin tilaosoittimen (prompt). Tilaosoittimesta voi päätellä, mikä ohjelma on kutsunut GETLNiä. Asteriski osoittimena tarkoittaa System Monitoria, oikea nuoli (>) viittää Apple Integrer BASICiin, oikea hakasulku ( ]) on Applesoft II BASICin osoitin ja huutomerkki (!) merkitsee Apple Mini-Assembleria. Lisäksi monet ohjelmat ja kielet käyttävät kysymysmerkkiä ilmaisemaan, että ohjelma tahtoo syöttötietoja. Käyttäjän näkökulmasta katsottuna Apple yksinkertaisesti kirjoittaa syöttö- ja tilaosoittimet (kursori ja prompt). Kun RETURNia painetaan, koko rivi lähetetään ajossa olevalle ohjelmalle tai kielelle, ja uusi prompt tulee näkyviin.

Mitä sitten itse asiassa tapahtuu, kun prompt on kirjoitettu? GETLN kutsuu RDKEYtä, joka asettaa syöttökursorin näkyviin. Kun RDKEY tuo näppäinkoodin, GETLN varastoi sen syöttöpuskuriin ja tulostaa sen näyttöön siihen kohtaan, missä syöttökursori oli. Sitten se kutsuu RDKEYtä uudestaan. Tätä jatkuu, kunnes käyttäjä painaa RETURNia. Kun GETLN saa RETURN-koodin näppäimistöltä, se sijoittaa sen syöttöpuskurin perään, tyhjentää sen rivin loppuun, jolla syöttökursori oli ja lähettää RETURN-koodin COUTille (katso edellä). Sen jälkeen GETLN palaa ohjelmaan, joka kutsui sitä. Nyt inputtia pyytänyt ohjelma tai kieli voi katsella koko riviä kerralla.

Milloin tahansa riviä kirjoitettaessa on mahdollista kirjoittaa CTRL X ja peruuttaa koko rivi. GETLN yksinkertaisesti unohtaa kaiken, mitä oli kirjoitettu, kirjoittaa vinoviivan (\ ), siirtyy uudelle riville ja kirjoittaa promptin sallien täten rivin kirjoittamisen uudelleen. Edelleen, GETLN voi käsitellä korkeintaan 255 merkkiä rivillä. Jos tämä raja ylitetään, GETLN unohtaa koko rivin ja käyttäjän on aloitettava uudelleen alusta. Varoituksena rajan lähestymisestä GETLN tulostaa äänimerkin jokaisella näppäimenpainalluksella alkaen 249:nnestä merkistä.

GETLN sallii myös kirjoitettavana olevan rivin korjailun ja muuttelun yksinkertaisten oikeinkirjoitusvirheiden korjaamiseksi. Lyhyt esittely tavallisista korjailumahdollisuuksista ja kahden

nuolen, < ja > , käytöstä löytyy sivuilta 28–29 ja 53–55 Apple II BASIC Programming Manuaalista tai sivuilta 27–28, 52–53 ja liite C:stä Applesoft Tutorialista. Ainakin toisen näistä olisi pitänyt seurata Applen mukana. Tässä seuraa lyhyt kuvaus GETLNin korjailu ominaisuuksista:

PERUUTUSMERKKI ([←])

Jokainen peruutusmerkin painallus saa GETLNin »unohtamaan» edellisen merkin syöttörivillä. Se myös lähettää peruutusmerkin COUTille (ks. edellä), jolloin kursori siirtyy hävitetyn merkin paikalle. Tällöin syötetty merkki korvaa hävitetyn merkin sekä näytössä että syöttörivillä. Useampi peruutusmerkki hävittää peräkkäisiä merkkejä. Jos peruutusmerkkejä kuitenkin tulee enemmän kuin kirjaimia oli alun perin kirjoitettu, GETLN unohtaa koko rivin ja kirjoittaa uuden promptin.

UUDELLEENKIRJOITUS ([→])

Uudelleenkirjoitusmerkillä on täsmälleen sama vaikutus kuin jos kirjoittaisi kursorin alla olleen merkin. Se on erittäin kätevä kun halutaan kopioida loppurivi jonka yli ollaan palattu korjaamaan kirjoitusvirhettä. Yhdessä tavallisten kursorin siirtojen (seuraava kappale) kanssa se on kätevä uudelleenkirjoitettaessa ja korjailtaessa jo näytössä olevia tietoja.

ESCAPE-KOODIT

Kun [ESC]-näppäintä painetaan, Applen syöttöalirutiinit siirtyvät escape-tilaan. Tässä tilassa yhdellätoista näppäimellä on erilliset merkityksensä ja niitä kutsutaan escape-koodeiksi. Kun yhtä näistä näppäimistä painetaan, Apple suorittaa siihen liitetyn toiminnon. Suorittamisen jälkeen Apple joko jatkaa escape-tilassa tai päättää sen, riippuen siitä, mitä näppäintä on painettu. Jos jotain sellaista näppäintä joka ei ole escape-koodi painetaan, ei siihen reagoida mitenkään ja escape-tila kumotaan.

Apple tuntee yksitoista escape-koodia, joista kahdeksan on tavallisia kursorin siirtoja, jotka vain siirtävät kursoria muuttamatta näyttöä tai syöttöriviä, ja muut kolme ovat näytön tyhjennyskoodeja, jotka tyhjentävät osan näytöstä. Kaikki näytöntyhjennyskoodit ja neljä ensimmäistä tavallista kursorinsiirtoa (escapekoodit @ , A, B, C, D, E, F) päättävät escape-tilan suoritettuaan tehtävänsä. Neljä jäljelle jäävää escapekoodia (I, K, M, J) taas jättävät escape-tilan aktiiviseksi.*

[ESC] [A] — [ESC]-näppäimen painallus, jota seuraa [A] kirjaimen painallus siirtää kursoria askeleen oikealle muuttamatta syöttöriviä. Tätä voidaan käyttää hypättäessä sellais-

* Näitä neljää escape-koodia ei ole Appleissa, joista puuttuu Autostart Monitor ROM.

ten merkkien yli, joita ei haluta syöttöriviin: peruutetaan ao. merkkien yli, ja sen jälkeen hypätään ei-toivottujen merkkien yli [ESC] [A] :lla ja muiden yli uudelleenkirjoitusnäppäimellä.

[ESC] [B] ESC-näppäimen painallus, jota seuraa [B] siirtää kursoria askeleen takaisin päin muuttamatta syöttöriviä. Tätä voidaan käyttää kun halutaan kirjoittaa jotakin kahdesti: kirjoitetaan se kerran, toistetaan [ESC] [B] , kunnes päästään lauseen alkuun ja kirjoitetaan se uudelleen pelkästään uudelleenkirjoitusnäppäimellä.

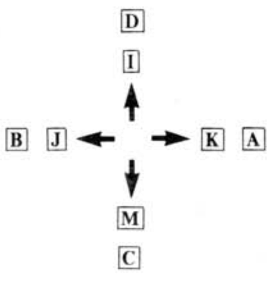
[ESC] [D] [ESC] [D] siirtää kursoria rivin verran ylöspäin ilman vaakasuoraa liikettä. Jos kursori saavuttaa ikkunan yläreunan, se ei liiku. Syöttörivi ei muutu. Koodilla voidaan siirtyä ylemmän rivin alkuun ja kirjoittaa se uudelleen uudelleenkirjoitusnäppäimellä.

[ESC] [E] Tätä koodia kutsutaan rivinselvittäjäksi. Kun COUT huomaa sen, se tyhjentää loput näyttörivistä (ei syöttörivistä) kursorista oikeaan laitaan asti. Kursori säilyy siinä missä on ja syöttörivi pysyy muuttumattomana. [ESC] [E] puhdistaa loppurivin aina blankoiksi riippumatta Normaalista/Käänteisestä tekstitilasta.

[ESC] [F] Tämä koodi tyhjentää kaiken, mitä on kursorin alapuolella tai samalla rivillä kursorin jälkeen. Kuten edellisissä, kursori ei nytkään liiku ja syöttörivi pysyy muuttumattomana. Tällä käskyllä voidaan mukavasti poistaa satunnaiset merkit näytöstä esim. korjailun ja useiden kursorinsiirtojen jälkeen.

[ESC] [@] Tämä käsky tyhjentää koko ikkunan ja asettaa kursorin vasempaan yläkulmaan. Taulu tyhjennetään blankoiksi riippumatta Normaalista/Käänteisestä tekstitilasta ja syöttörivi ei muutu. (huom! [@] = [SHIFT P]).

[ESC] [K]
[ESC] [J]
[ESC] [M]
[ESC] [I]
Nämä neljä escape-koodia ovat edellä mainittujen tavallisten kursorinsiirtojen synonyymeja. Kun ne ovat suorittaneet tehtävänsä, ne eivät käännä escape-tilaa pois päältä. Näitä escape-koodeja voi painella edelleen ja siirrellä kursoria ympäri näyttöä kunnes painetaan jotain muuta kuin ESC-koodia. Nämä neljä koodia voidaan merkitä suuntanuolella niin, että jokaista näppäintä osoittava nuoli näyttää suunnan, johon vastaava koodi siirtää kursoria näytössä.



Kuva 4. Kursorin liikuttelun escape-koodit

NOLLAUSRUTIINI (RESET)

Kun Applen virta kytketään päälle* tai painetaan RESET näppäintä, aloittaa 6502 mikroprosessori NOLLAUS (RESET) -rutiinin. Se alkaa hyppäämällä Monitor ROMin alirutiiniin. ROMin kahdessa eri versiossa, Monitor ROMissa ja Autostart ROMissa, RESET-rutiini tekee eri asioita.

AUTOSTART ROM-RESET

Applet, joilla on Autostart ROM, aloittavat RESET-rutiinin kääntämällä ne ohjelmalliset kytkimet, jotka siirtävät näytöön tekstitilan ensimmäisen sivun. Perusgrafiikka-sekatila piileskelee tekstin takana. Sitten teksti-ikkuna avataan täyteen kokoonsa, tulostuskursori pudotetaan näytön alalaitaan ja normaalitila kytketään päälle. Sen jälkeen COUT- ja KEYIN -kytkimet asetetaan niin, että Applen näppäimistöä ja kuvaruutua voidaan käyttää perus I/O-välineinä. Sitten käännetään pelien I/O-liittimen kutsu-ulostulot Ø ja 1 ON ja kutsu-ulostulot 2 ja 3 OFF. Nollataan näppäimistön tilailmaisin (STROBE), käännetään kaikki I/O-lisäliitännät pois päältä (ks. sivu 84) ja päästetään »piip»-ääni.

Nämä toimet suoritetaan joka kerta, kun RESET näppäintä painetaan. Tällöin Autostart ROM tutkii kaksi tiettyä muistipaikkaa nähdäkseen, onko RESET-rutiini suoritettu aiemmin vai onko Apple vasta käynnistetty (nämä muistipaikat käsitellään tarkemmin myöhemmin). Jos Apple on vasta käynnistetty, Autostart ROM suorittaa »kylmän aloituksen», muulloin »lämpimän aloituksen».

1. **Kylmä aloitus.** Vastakäynnistetyssä Applessa RESET-rutiini jatkuu näytön tyhjentämisellä kirjoittamalla teksti »APPLE II» näytön yläreunaan. Sen jälkeen se merkitsee tiettyihin muistipaikkoihin, että se on käynnistetty ja RESET-rutiini suoritettu. Tämän jälkeen se käy läpi seitsemän oikeanpuolimmaisinta väylää Applen takaosassa ja etsii Disk II:n ohjainkorttia. Se aloittaa etsinnän väylästä 7 ja jatkaa sitä väylään 1. Jos se löytää liitäntäkortin, se lataa automaattisesti Applen levynkäyttöjärjestelmän (DOS) levyltä, joka on ko. liitäntäkortin asemassa 1. Tästä toiminnasta löytyy tarkempi kuvaus Applen levynkäyttöjärjestelmää käsittelevästä kirjasta: DOS' and DON'TS of DOS, Apple-tuote n:o A2T0012, sivulta 11.

   Jos Autostart ROM ei löydä Disk II:n ohjainlevyä, tai RESET iä painetaan ennen kuin DOS on täysin ladattu, RESET-rutiini jatkaa »haalealla aloituksella». Se asettaa osoittimet ja hyppää kieleen, joka on asennettu Applen lukumuisteihin (ROM). Ilman Applesoft II Firmware -levyä tai jos ko. kortin ohjauskytkin on ALAS-asennossa, Autostart ROM aloittaa Apple Integer BASICin. Apple II Plus-järjestelmissä tai Version 0 Appleissa, joissa on Applesoft II Firmware levy ja ktykin YLÄ-asennossa, Autostart ROM aloittaa Applesoft II liukuvan pisteen desimaali BASICin.

2. **Lämmin aloitus.** Jos Autostart ROM on jo suorittanut kylmän aloituksen, niin joka kerta RESET iä painettaessa se palauttaa käyttäjän ao. kieleen, ohjelmat ja muuttujat koskemattomina.

---

* Käynnistys-RESET-rutiini tapahtuu vain version 1 Appleilla tai niillä Version Ø Appleilla, joilla on vähintään yksi Disk II ohjainlevy.

AUTOSTART ROMIN ERIKOISMUISTIPAIKAT

Autostart ROMin käyttämät kolme »erikoismuistipaikkaa» sijaitsevat kaikki RAM-muistin alueella, joka on varattu näille erikoistoiminnoille. Seuraavassa taulukossa ovat Apple Autostart ROMin käyttämät muistipaikat:

Taulukko 13. Autostart ROMin erikoismuistipaikat

Muistipaikka Desim.	Hex	Sisältö:
1010 1011	$3F2 $3F3	Paluuvektori (Soft entry vector). Muistipaikat sisältävät paluuosoitteen kulloin käytössä olevaan kieleen. Tavallisesti $E003.
1012	$3F4	Käynnistystavu. Normaali sisältö $45. Katso alla.
64367 (-1169)	$FB6F	Alkuosoite konekieliseen ohjelmaan, joka asettaa käynnistystavun.

Kun Applen virta kytketään, Autostart ROM asettaa tietyn arvon »virta-päälle»-muistipaikkaan. Tämä on eksklusiivinen Ex-OR arvo arvosta, joka on muistipaikassa 1011 ja vakiosta 165. Jos esimerkiksi muistipaikassa 1011 on arvo 224 (sen normaali arvo), »virta-päälle»-arvoksi tulee:

	Desim.	Hex	Binaari
Muistipaikka 1011	224	$E0	11100000
Vakio	165	$A5	10100101
Käynnistysarvo	69	$45	01000101

Ohjelma voi muuttaa näitä arvoja niin, että RESETiä painamalla joudutaan johonkin muuhun ohjelmaan eikä käytettyyn ohjelmointikieleen, kuten normaalisti. Jos tähän vektoriin kajotaan, on syytä varmistaa, että »virta-päälle»-tavu vektorin eniten merkitsevästä osasta täsmää desimaaliluvun 165 (heksadesimaali $A5 kanssa). Muussa tapauksessa seuraavan RESETin yhteydessä Autostart ROM luulee, että Apple on vasta käynnistetty ja suorittaa uuden kylmän aloituksen.

Muuttakaamme vektoria esimerkiksi niin, että RESETin painaminen johtaa Monitoriin. Tämän muutoksen suorittamiseksi Monitori-rutiinin aloitusmuistipaikan osoite täytyy sijoittaa »virta-päälle»-vektoriin. Monitori alkaa osoitteesta $FF69 (tai desimaali 65385). Osoitteen kaksi viimeistä heksadesimaalimerkkiä ($69) sijoitetaan osoitteeseen $3F2 ja kaksi ensimmäistä merkkiä osoitteseen $3F3). Jos toiminta tapahtuu desimaaliluvuilla, sijoitetaan 105 (jakojäännös laskusta 65385/256) osoitteeseen 1010 ja 255 (kokonaisosa laskusta 65385/256) muistipaikkaan 1011.

Nyt on vielä järjestettävä »virta-päälle»-muistipaikka. Autostart ROMissa on konekielinen alirutiini, joka automaattisesti asettaa sen yllä mainitun eksklusiivisen OR-funktion mukaiseksi. On vain suoritettava hyppykäsky (JSR) muistipaikkaan $FB6F. Jos toimitaan BASICissa, on suoritettava käsky CALL -1169. Nyt kaikki on valmista ja seuraavalla RESETillä päädytään System Monitoriin.

Kun RESET halutaan saada toimimaan taas normaalilla tavallaan, palautetaan vektorin alkuperäiset arvot ($E003 tai desimaali 57347) ja kutsutaan edellä mainittua alirutiinia.

»VANHA MONITOR»-ROMIN RESET-RUTIINI

RESET-rutiini Apple II Monitor ROMissa tapahtuu siten, että asetetaan normaali videotila päälle, kytketään näytöksi tekstisivu 1, värigrafiikka-sekatila sen takana, ja asetaan Applen näppäimistö ja näyttö standardi-I/O -välineiksi. Sitten kuuluu »piip»-ääni, kursori sijoitetaan tyhjentämättömän näytön alareunaan ja käyttäjä saa eteensä asteriski-tilailmaisimen (prompt) ja huomaa puhuvansa Apple System Monitorille.

LUKU 3
SYSTEM MONITOR

Syvällä ApplenROMin kätköissä on ohjelmointitekninen mestariteos nimeltä System Master. Se toimii sekä systeemin valvojana että sen palvelijana; se valvoo kaikkia ohjelmia ja kaikki ohjelmat voivat käyttää sitä. System Monitorin voimalla voi tutkia kaikkien 65.536:n muistipaikan sisällön; monitorista voidaan katsella yhtä, muutamaa tai kaikkia muistipaikkoja; minkä tahansa muistipaikan sisältö voidaan muuttaa; konekieliset ja Assemblerohjelmat voidaan kirjoittaa suoraan Applen mikroprosessorin suoritettaviksi; valtavia määriä data-tietoa ja ohjelmia voidaan varastoida kasetille ja lukea jälleen takaisin; tuhansia tavuja muistia voidaan siirtää ja vertailla yhdellä käskyllä ja lisäksi voidaan jättää monitori ja siirtyä mihin tahansa muuhun ohjelmaan.

MONITORIIN PÄÄSEMINEN

Apple System Monitor -ohjelma alkaa muistipaikasta $FF69 (desimaali 65385 tai -151). Monitoriin pääsemiseksi Sinun tai BASIC-ohjelmasi on kutsuttava tätä muistipaikkaa. Monitorin tilailmaisin (prompt) (asteriski [*]) ilmaantuu näytön vasempaan laitaan vilkkuva kursori oikealla puolellaan. Monitori hyväksyy standardimuotoiset syöttörivit (ks. sivu 32) aivan kuin mikä tahansa muukin Apple-systeemi. Se ei ryhdy mihinkään ennen RETURN in painamista. Syöttörivit saavat olla korkeintaan 255 merkkiä pitkiä. Kun haluat siirtyä pois Monitorista, voit palata aiemmin käyttämääsi kieleen kirjoittamalla CTRL C RETURN (tai, jos Apple DOS on käytössä, 3 D Ø G RETURN tai painamalla RESET.*

OSOITTEET JA DATA-TIEDOT

Monitorille puhuminen on samanlaista kuin puhuminen jollekin muulle Applen ohjelmalle tai kielelle: Kirjoitat näppäimistön avulla rivin, jota seuraa RETURN ja Monitor järjestelee kirjoittamasi aineiston ja toimii saamiensa ohjeiden mukaan. Monitorille voit antaa kolmen tyyppistä tietoa: osoitteita, arvoja ja käskyjä. Osoitteet ja arvot annetaan Monitorille heksadesimaalijärjestelmässä. Heksadesimaalijärjestelmässä kymmenen desimaalimerkkiä (0–9) ovat samat kuin kymmenjärjestelmässäkin ja ensimmäiset kuusi kirjainta (A–F) vastaavat numeroita 10–15. Yhden heksadesimaalimerkin arvo voi siis vaihdella 0:sta 15:een. Heksadesimaalimerkkiparin arvo voi olla välillä 0–255 ja kuuden merkin ryhmä voi tarkoittaa mitä tahansa numeroa välillä 0–65.535. Sattuu olemaan niin, että mikä tahansa Applen osoite voidaan ilmaista neljällä merkillä ja mikä tahansa arvo kahdella merkillä. Näin Monitorille annetaan osoitteet ja arvot. Kun Monitori etsii osoitetta, se hyväksyy minkä tahansa heksadesimaalimerkkiryhmän. Jos ryhmässä on vähemmän kuin neljä merkkiä, se lisää alkuun nollia; jos merkkejä on enemmän kuin neljä, se käyttää vain neljä viimeistä merkkiä. Samalla lailla se ottaa sisään kahden merkin datan.

Monitor tunnistaa 22 eri käskymerkkiä. Jotkut niistä ovat välimerkkejä, jotkut isoja kirjaimia tai kontrollikirjaimia. Seuraavassa kappaleessa käskyn koko nimi kirjoitetaan isoilla kirjaimilla. Monitor tarvitsee kuitenkin vain nimen ensimmäisen kirjaimen. Jotkut käskyt annetaan kontrollikirjaimilla. On syytä huomata, että vaikka Monitor tunnistaa ja hyväksyy kontrollikirjaimet, ne eivät tule näkyviin näytössä.

* Tämä ei onnistu, jos Autostart ROM puuttuu.

Monitor muistaa osoitteet viiteen muistipaikkaan asti. Kaksi näistä ovat erikoisia: Ne ovat muistipaikka, jonka arvoa on viimeksi tiedusteltu ja muistipaikka, jonka arvo seuraavaksi muutetaan. Näitä kutsutaan *viimeiseksi avatuksi* ja *seuraavaksi muutettavaksi* muistipaikaksi. Näiden kahden osoitteen käyttökelpoisuus tulee ilmi pian.

MUISTIN SISÄLLÖN TUTKIMINEN

Kun kirjoitat ainoastaan muistipaikan osoitteen Monitorin syöttöriville, se vastaa kirjoittamalla antamasi osoitteen, yhdysviivan ja muistipaikan sisältämän arvon** seuraavasti:

```
*E000

E000- 20
*300

0300- 99
*
```

Joka kerta kun Monitor näyttää jonkin muistipaikan arvon, se muistaa tämän osoitteen *viimeisenä avattuna muistipaikkana*. Teknisistä syistä se muistaa tämän myös *seuraavana muutettavana muistipaikkana*.

TUTKITAAN LISÄÄ MUISTIA

Jos kirjoitat Monitorin syöttöriville pisteen (.), jota seuraa osoite, Monitori näyttää muistilistauksen: kaikkien niiden muistipaikkojen arvot, jotka sijaitsevat viimeksi avatun muistipaikan ja pisteen perään kirjoittamasi muistipaikan välillä. Sen jälkeen Monitor siirtää viimeisen näytetyn muistipaikan sekä viimeiseksi avatuksi että seuraavaksi muutettavaksi osoitteeksi.

* Esimerkeissä Sinun tekstisi on normaalia ja Applen vastaukset on ladottu lihavalla.

** Näissä esimerkeissä näkyvät arvot voivat poiketa niistä, jotka Sinun Applesi esittää.

```
*20

0020- 00
*.2B

0021- 28 00 18 0F 0C 00 00
0028- A8 06 D0 07
*300

0300- 99
*.315

0301- B9 00 08 0A 0A 0A 99
0308- 00 08 C8 D0 F4 A6 2B A9
0310- 09 85 27 AD CC 03
*.32A

0316- 85 41
0318- 84 40 8A 4A 4A 4A 4A 09
0320- C0 85 3F A9 5D 85 3E 20
0328- 43 03 20
*
```

Muistin listauksen (memory dump) muodosta voi tehdä monia huomioita: Ensiksi, listauksen ensimmäinen rivi alkaa viimeksiavattua osoitetta seuraavalla osoitteella. Toiseksi, kaikki rivit alkavat osoitteilla, jotka päättyvät joko nollaan tai kahdeksikkoon. Kolmanneksi, muistin listauksen rivillä ei ole koskaan enempää kuin kahdeksan arvoa. Kun Monitor tulostaa listauksen, se aloittaa näyttämällä viimeista avattua muistipaikkaa seuraavan muistipaikan arvon. Sen jälkeen se jatkaa esittämällä seuraavan arvon jne. Jos muistipaikan osoite päättyy kahdeksikkoon tai nollaan, Monitor »katkaisee» rivin ja sijoittaa osoitteen arvoineen seuraavan rivin alkuun ja jatkaa arvojen esittämistä. Kun se on esittänyt määrittämäsi muistipaikan arvon, se lopettaa muistin listauksen ja asettaa sen viimeisen osoitteen sekä viimeiseksi avatuksi että ensimmäiseksi muutettavaksi osoitteeksi. Jos syöttörivillä määritelty osoite on pienempi kuin viimeksi avattu osoite, Monitor näyttää vain viimeistä avattua osoitetta seuraavan muistipaikan osoitteen ja arvon.

Nämä kaksi käskyä (avaus ja listaus) voidaan yhdistää yhdeksi operaatioksi liittämällä ne peräkkäin, mikä tarkoittaa: kirjoita ensimmäinen osoite, sen jälkeen piste ja sitten toinen osoite. Tätä kahden pisteellä erotetun osoitteen yhdelmää sanotaan muistialueeksi.

```
*300.32F

0300- 99 B9 00 08 0A 0A 0A 99
0308- 00 08 C8 D0 F4 A6 2B A9
0310- 09 85 27 AD CC 03 85 41
0318- 84 40 8A 4A 4A 4A 4A 09
0320- C0 85 3F A9 5D 85 3E 20
0328- 43 03 20 46 03 A5 3D 4D
*30.40

0030- AA 00 FF AA 05 C2 05 C2
0038- 1B FD D0 03 3C 00 40 00
0040- 30
*E015.E025
```

```
E015- 4C ED FD
E018- A9 20 C5 24 B0 0C A9 8D
E020- A0 07 20 ED FD A9
*
```

TUTKITAAN VIELÄ LISÄÄ MUISTIA

Pelkän RETURNin painaminen saa Monitorin vastaamaan yhdellä muistilistauksen rivillä. Se on viimeksiavattua muistipaikkaa seuraava rivi lähimpään kahdeksikkoon tai nollaan asti. Jälleen kerran viimeksi näytetty osoite valitaan viimeiseksi avatuksi ja seuraavaksi muutettavaksi osoitteeksi.

```
*5

0005- 00
*[RETURN]
 00 00
*[RETURN]

0008- 00 00 00 00 00 00 00 00
*32

0032- FF
*[RETURN]
 AA 00 C2 05 C2
*[RETURN]

0038- 1B FD D0 03 3C 00 3F 00
*
```

MUISTIPAIKKOJEN SISÄLLÖN MUUTTAMINEN

Aikaisemmissa kappaleissa olemme puhuneet useasti »seuraavasta muutettavasta muistipaikasta». Nyt pääset käsittelemään sitä. Kirjoita kaksoispiste, jota seuraa arvo.

```
*0

0000- 00
*:5F
```

Yllätys! Seuraavan muutettavan muistipaikan arvo on juuri muuttunut kirjoittamaksesi arvoksi. Tarkasta asia tutkimalla muistipaikka uudelleen:

```
*0

0000- 5F
```

```
*
```

Voit myös yhdistää avaamisen ja vaihtamisen samaan operaatioon.

```
*302:42

*302

0302- 42
*
```

Kun muutat muistipaikan sisällön, sen sisältämä vanha arvo katoaa, eikä sitä saa millään takaisin. Uusi arvo säilyy, kunnes se korvataan toisella heksadesimaaliluvulla.

PERÄKKÄISTEN MUISTIPAIKKOJEN SISÄLLÖN MUUTTAMINEN

Sinun ei tarvitse kirjoittaa osoitetta, kaksoipistettä ja RETURNia joka ainoalle muistipaikalle, jonka haluat muuttaa. Monitor antaa mahdollisuuden muuttaa kerralla jopa kahdeksankymmentäviisi muistipaikkaa kirjoittamalla ensimmäisen osoite, kaksoispiste ja sitten kaikki arvot blankoilla (välilyönneillä) eroteltuina. Käsiteltyään tämän jonon, Apple ottaa viimeksi muutettua osoitetta seuraavan muistipaikan seuraavaksi muutettavaksi osoitteeksi. Jos siis haluat jatkaa peräkkäisten muistipaikkojen sisällön muuttamista, voit kirjoittaa seuraavan rivin alkuun kaksoispisteen ja jatkaa arvojen kirjoittamista.

```
*300:69 01 20 ED FD 4C 0 3

*300

0300- 69
*[RETURN]
 01 20 ED FD 4C 00 03
*10:0 1 2 3

*:4 5 6 7

*10.17

0010- 00 01 02 03 04 05 06 07
*
```

MUISTIALUEEN SIIRTÄMINEN

Voit käsitellä muistialuetta (kaksi osoitetta, joiden välissä on piste) kokonaisuutena ja siirtää sen

paikasta toiseen käyttämällä Monitorin MOVE-käskyä. Jos muistialue halutaan siirtää paikasta toiseen, Monitorille täytyy kertoa sekä missä alue muistissa sijaitsee että minne se on siirrettävä. Tämä informaatio annetaan Monitorille kolmessa osassa: muistialueen määränpään osoite, musitialueen ensimmäisen muistipakan osoite sekä muistialueen viimeisen muistipaikan osoite. Muistialueen alku- ja loppuosoitteet määritellään normaaliin tapaan erottamalla ne pisteellä. Tämä alue erotetaan määränpäästä vasemmalla nuolella (<). Lopuksi kerrot Monitorille, että haluat siirtää tämän alueen kirjoittamalla M-kirjaimen (sanasta MOVE). Lopullinen käsky näyttää seuraavalta:

{määränpää} < {alku}.{loppu} M

Kun kirjoitat tämän Monitorille, aaltosuluissa olevat korvataan luonnollisesti heksadesimaaliosoitteilla ja blankot jätetään pois. Seuraavassa muutamia esimerkkejä siirroista:

```
*Ø.F

ØØØØ- 5F ØØ Ø5 Ø7 ØØ ØØ ØØ ØØ
ØØØ8- ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
*3ØØ:A9 8D 2Ø ED FD A9 45 2Ø DA FD 4C ØØ Ø3

*3ØØ.3ØC

Ø3ØØ- A9 8D 2Ø ED FD A9 45 2Ø
Ø3Ø8- DA FD 4C ØØ Ø3
*Ø<3ØØ.3ØCM

*Ø.C

ØØØØ- A9 8D 2Ø ED FD A9 45 2Ø
ØØØ8- DA FD 4C ØØ Ø3
*31Ø<8.AM

*31Ø.312

Ø31Ø- DA FD 4C
*2<7.9M

*Ø.C

ØØØØ- A9 8D 2Ø DA FD A9 45 2Ø
ØØØ8- DA FD 4C ØØ Ø3
*
```

Monitor tekee yksinkertaisesti kopion siirrettävästä alueesta ja siirtää sen määränpäähän. Alkuperäinen alue säilyy muuttumattomana. Monitor muistaa alkuperäisen alueen viimeisen muistipaikan viimeksi avattuna muistipaikkana ja alkuperäisen alueen ensimmäisen muistipaikan seuraavana muutettavana muistipaikkana. Jos muistialueen toinen osoite on ensimmäistä pienempi, vain yksi arvo (alueen ensimmäisen muistipaikan arvo) siirretään.

Jos MOVE-käskyä määränpääalue on siirrettävän alueen sisällä, kummallisia ja (joskus) varsin mukavia asioita tapahtuu: Muistialueen alun ja määränpään alun välillä olevia muistipaikkoja käsitellään alialueena ja tämän alialueen arvot sijoitetaan yhä uudelleen alkuperäiselle muistialueelle. Katso »Monitoritrikkejä», sivu 55, jossa on tämän ominaisuuden kiintoisa sovellutus.

KAHDEN MUISTIALUEEN VERTAILU

Monitorin käyttö kahden muistialueen vertailussa tapahtuu melkein samanmuotoisella käskyllä kuin muistialueen siirtäminen paikasta toiseen. Itse asiassa VERIFY-käskyä voidaan käyttää heti MOVE-käskyn jälkeen tarkistamaan, että siirto on oonistunut.

VERIFY-käsky, kuten MOVE-käskykin, tarvitsee muistialueen ja määränpään. Lyhyesti:

{ määränpää } { alku } . { loppu } V

Monitor vertaa tässä määriteltyä muistialuetta määränpääosoitteesta alkavaan muistialueeseen. Jos kaikki ei täsmää, Monitor kirjoittaa näkyviin osoitteen, jossa ero syntyi ja eroavat arvot.

```
*Ø:D7 F2 E9 F4 F4 E5 EE AØ E2 F9 AØ C3 C4 C5

*3ØØ<Ø.DM

*3ØØ<Ø.DV

*6:E4

*3ØØ<Ø.DV

ØØØ6-E4 (EE)
*
```

Huomaa, että VERIFY-käsky virheen löytäessään kirjoittaa alkuperäisen alueen muistipaikan, jonka arvo eroaa määränpääalueella olevasta arvosta. Jos eroja ei ole, mitään ei kirjoiteta näyttöön. Molemmat alueet pysyvät muuttumattomina. Viimeksi avattu ja seuraava muutettava osoite asetetaan kuten MOVE-käskyssä. Kuten edellä, jos muistialueen jälkimmäinen osoite on edellistä pienempi, vain alueiden ensimmäisiä arvoja verrataan. VERIFY tekee myös omintakeisia tempauksia, jos määränpää on alkuperäisen alueen sisällä, katso »Monitoritrikkejä», sivu 55.

MUISTIALUEEN TALTIOIMINEN NAUHALLE

Monitorilla on kaksi erikoiskäskyä, jotka sallivat muistialueen taltioimiseen kasetille ja sen purkamisen myöhempää käyttöä varten. Ensimmäinen näistä käskyistä, WRITE, antaa mahdollisuuden varastoida 1—65.536:n muistipaikan sisällöt nauhalle.

Taltioidaksesi muistialueen nauhalle annan Monitorille alueen alku- ja loppuosoitteet ja niiden perässä W-kirjain.

```
{alku} . {loppu} W
```

Jos haluat onnistuneen nauhoituksen, on syytä kytkeä nauhuri äänitykselle ennen RETURN in painamista. Anna nauhan juosta muutamia sekunteja ja paina sitten RETURN ia. Monitor kirjoittaa kymmenen sekunnin »otksikon» ja sen perään datatiedot. Kun monitor on lopettanut, se päästää »piip»-äänen ja kirjoittaa uuden promptin. Silloin tulisi kelata nauha takaisin ja kirjoittaa kasettiin tiedot nauhalla olevasta muistialueesta ja mitä sen on tarkoitus pitää sisällään.

```
*Ø.FF FF AD 3Ø CØ 88 DØ Ø4 C6 Ø1 FØ Ø8 C
A DØ F6 A6 ØØ 4C Ø2 ØØ 6Ø

*Ø.14

ØØØØ- FF FF AD 3Ø CØ 88 DØ Ø4
ØØØ8- C6 Ø1 FØ Ø8 CA DØ F6 A6
ØØ1Ø- ØØ 4C Ø2 ØØ 6Ø
*Ø.14W

*
```

4.096 muistipaikan varastointi kymmenen sekunnin otsikkoineen vaatii aikaa noin 35 sekuntia. Tämä tekee tallennuksen keskimnopeudeksi 1.350 bittiä sekunnissa. WRITE-käsky kirjoittaa yhden ylimääräisen luvun nauhalle kirjoitettuaan kaikki muistialueen luvut. Tämä ylimääräinen luku on tarkistussumma. Se muodostetaan kaikkien alueella olevien arvojen summasta. READ-alirutiini käyttää sitä tarkistaessaan, onko lukuoperaatio onnistunut (ks. alla).

MUISTIALUEEN LUKEMINEN NAUHALTA

Kun muistialue on varastoitu nauhalle Monitorin WRITE-käskyllä, se voidaan lukea takaisin Appleen Monitorin READ-käskyllä. Nauhalle varastoitua datatietoa ei tarvitse lukea takaisin samalle muistialueelle, mistä se oli lähtöisin. Monitoria voi käskeä sijoittamaan arvot mihin tahansa samankokoiseen muistialueeseen Applen muistiavaruudessa.

READ-käskyn muoto on sama kuin WRITE-käskyn, paitsi että käskykirjain on R eikä W:

```
{alku} . {loppu} R
```

Kun olet kirjoittanut käskyn, älä taaskaan paina heti RETURN ia. Kytke sen sijaan nauhuri toistolle ja anna nauhan ei-magneettisen osan mennä ohi. Vaikka WRITE-käsky sijoittaakin nauhalle kymmenen sekunnin otsikon, READ-käsky tarvitsee vain kolme sekuntia tätä otsikkoa lukkiutuakseen oikealle taajuudelle. Niinpä nauhan pitäisi antaa kulkea muutamia sekunteja ennen RETURN in painamista, jotta nauhurin ulostulo saisi tuotettua tasaisen huojumattoman äänen.

```
*Ø:Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø
Ø Ø

*Ø.14
```

```
0000- 00 00 00 00 00 00 00 00
0008- 00 00 00 00 00 00 00 00
0010- 00 00 00 00 00
*0.14R

*0.14

0000- FF FF AD 30 C0 88 D0 04
0008- C6 01 F0 08 CA D0 F6 A6
0010- 00 4C 02 00 60
*
```

Kun Monitor on lukenut ja siirtänyt kaikki nauhalla olleet luvut, se lukee vielä tarkistussumman. Se vertaa sitä itse saamaansa tarkistussummaan ja jos niissä on eroja, Monitor päästää »piip»-äänen kaiuttimella ja kirjoittaa näyttöön »ERR». Tämä on varoitus siitä, että lukemisen aikana oli ongelmia ja että luetut arvot eivät ole niitä, jotka alunperin varastoitiin nauhalle. Jos tarkistussummat taas täsmäävät, Monitori antaa uuden promptin.

KONEKIELISTEN OHJELMIEN LUOMINEN JA AJAMINEN

Konekieli on varamasti Applen tehokkain kieli, joskin se on myös ikävintä käyttää. Monitorilla on lisäapuneuvoja niille, jotka ovat vakaasti päättäneet käyttää konekieltä. Näillä voidaan yksinkertaistaa konekielisten ohjelmien luomista, kirjoittamista ja virheiden etsintää.

On mahdollista kirjoittaa konekielinen ohjelma, ottaa operandien ja op-koodien heksadesimaaliarvot ja varastoida ne muistiin edellä kuvatuilla käskyillä. Ohjelmasta voidaan saada heksadesimaalilistaus, sitä voidaan liikuttaa ympäri muistissa tai varastoida se nauhalle ja lukea sieltä takaisin. Kaikki tämä tapahtuu jo oppimillasi käskyillä. Kaikkein tärkein käsky konekielisten ohjelmien kanssa puuhattaessa on kuitenkin GO-käsky. Kun avaat muistipaikan Monitorissa ja kirjoitat G:n, Monitor saa 6502 mikroprosessorin aloittamaan viimeisestä avatusta muistipaikasta alkavan konekielisen ohjelman suorituksen. Monitori käsittelee ohjelmaa alirutiinina: kun ohjelma on päättynyt, sen tarvitsee vain suorittaa paluukäsky RTS (return from subroutine) siirtääkseen kontrollin takaisin Monitorille.

Konekieliset ohjelmat voivat kutsua useita Monitorin alirutiineita tekemään erilaisia tehtäviä. Tässä on esimerkki, jossa ladataan ja ajetaan konekielinen ohjelma, joka tulostaa kirjaimet A:sta Z:aan:

```
*300:A9 C1 20 ED FD 18 69 1 C9 DB D0 F6 60

*300.30C

0300- A9 C1 20 ED FD 18 69 01
0308- C9 DB D0 F6 60
*300G
ABCDEFGHIJKLMNOPQRSTUVWXYZ
*
```

(Apple 6502 mikroprosessorin käskyvalikoima on esitetty tämän käsikirjan liitteessä A.)

Suora heksadesimaalikoodi ei kuitenkaan ole maailman helpoin tapa lukea tai korjailla. Tätä silmälläpitäen Applen luojat ovat liittäneet mukaan käskyn, jolla konekieliset ohjelmat voi listata assembly-kielisinä. Tämä merkitsee sitä, että yhden, kahden tai kolmen tavun muodostaman sekavan heksadesimaalikielen asemesta käskyt tulevat kolmikirjaimisina, muistikkaina (mnemonics) sekä hieman järjestettyinä heksadesimaali-ilmaisuina. Monitorin LIST-käsky aloittaa tietystä muistipaikasta ja antaa käskyjä näytöllisen kerrallaan (20 riviä).

```
*300L

0300-   A9 C1       LDA   #$C1
0302-   20 ED FD    JSR   $FDED
0305-   18          CLC
0306-   69 01       ADC   #$01
0308-   C9 DB       CMP   #$DB
030A-   D0 F6       BNE   $0302
030C-   60          RTS
030D-   00          BRK
030E-   00          BRK
030F-   00          BRK
0310-   00          BRK
0311-   00          BRK
0312-   00          BRK
0313-   00          BRK
0314-   00          BRK
0315-   00          BRK
0316-   00          BRK
0317-   00          BRK
0318-   00          BRK
0319-   00          BRK
*
```

Tuntuvatko ensimmäiset rivit tutuilta? Siinä on assebly-kielinen koodi pari sivua sitten kirjoittamastasi ohjelmasta. Loput riveistä (BRK-käskyt) ovat vain näytön täytteenä. Määrittelemäsi osoite on koneen muistissa, mutta ei millän edellä selitetyllä tavalla. Se sijoitetaan ohjelmalaskuriin, jota käytetään pelkästään muistipaikkojen osoittamiseen ohjelman sisällä. LIST-käskyn jälkeen ohjelmalaskuri asetetaan soittamaan heti viimeksi näytettyä osoitetta seuraavaa muistipaikkaa, joten seuraava LIST-komento tuottaa uuden taulullisen ohjeita alkaen siitä, mihin edellinen lopetti.

MINI-ASSEMBLER

Monitorissa on toinen ohjelma*, joka antaa mahdollisuuden kirjoittaa Apple-ohjelmia samassa asseblymuodossa kuin mikä esiintyy LIST-käskyssä. Tätä ohjelmaa kutsutaan Apple Mini-Assembleriksi. Se on »mini»-assembler, koska se ei ymmärää symbolimerkintöjä, mihin oikean assemblerin täytyy pystyä. Mini-Assemblerin ajamiseksi kirjoita:

---

* Mini-Assembler ei oikeastaan ole Monitor ROMissa vaan on osa Apple Integer BASIC ROMia. Niinpä sitä ei voi käyttää Apple Plus -systeemeissä, tai kun Firmware Applesoft II on käytössä.

*F666G

!

Nyt olet Mini-Assemblerissa. Se käyttää promptina huutomerkkiä. Mini-Asseblerissa ollessasi voit suorittaa Monitor-käskyn kirjoittamalla sen eteen dollarimerkin ($). Lisäksi Mini-Assemblerilla on oma käskyvalikoimansa ja kielioppi.

Mini-Assembler muistaa yhden osoitteen, nimittäin ohjelmalaskurin. Ennen kuin aloitat ohjelman syöttämisen, Sinun täytyy kirjoittaa sen muistipaikan osoite, josta haluat ohjelman alkavan. Tämä tehdään kirjoittamalla osoite ja sen perään kaksoispiste. Kirjoita tämän jälkeen ensimmäisen käskyn muistikas ja sen perään blankko. Kirjoita nyt käskyn operandi (operandien muodot ovat sivulla 66). Nyt voit painaa RETURN ia. Mini-Assembler muuttaa kirjoittamansa rivin heksadesimaaleiksi, varastoi sen muistiin ohjelmalaskurin osoittamaan muistipaikkaan. Tämän jälkeen se purkaa heksadesimaalikoodin jälleen muistikkaaksi ja kirjoittaa heksadesimaalimuodon ja puretun muodon syöttämäsi rivin perään. Sen jälkeen se kirjoittaa promptin seuraavalle riville. Kertokasesi, että haluat seuraavan käskyn seuraavan edellistä, älä kirjoita osoitetta tai kaksoispistettä vaan blankko, jonka perään seuraavan käskyn muistikas ja operandi. Paina RETURN ia. Rivi otetaan sisään ja kieli jää odottamaan seuraavaa.

Jos kirjoittamassasi rivissä on virhe, Mini-Assebler piippaa ja kirjoittaa ylös osoittavan nuolenpään (circumflex) (^) syöttörivillä olevan epäilyttävän merkin alle tai lähelle. Tavallisimmat virheet ovat oikeinkirjoitusvirheitä: väärin kirjoitettuja muistikkaita, puuttuvia sulkuja jne. Mini-Assembler hylkää syöttörivin myös jos olet unohtanut blankon muistikkaan perästä tai heksadesimaaliluvussa tai osoitteessa on ylimääräisiä merkkejä. Jos hyppykäskyn (BRANCH) määräosoite on rajojen ulkopuolella (kauempana kuin 127 osoitteen päässä käskystä), Mini-Assembler katsoo tämänkin virheeksi.

```
!300:LDX #02

0300-    A2 02        LDX    #$02
! LDA $0,X

0302-    B5 00        LDA    $00,X
! STA $10,X

0304-    95 10        STA    $10,X
! DEX

0306-    CA           DEX
! STA $C030

0307-    8D 30 C0     STA    $C030
! BPL $302

030A-    10 F6        BPL    $0302
! BRK

030C-    00           BRK
!
```

Kun haluat päästä Mini-Assemblerista takaisin Monitoriin, paina joko RESET iä tai kirjoita

Monitori-käsky (jota edeltää dollarimerkki) FF69G:

```
!$FF69G

*
```

Assembly-ohjelmasi on nyt varastoituna musitiin. Voit tarkastaa asian LIST-komennolla:

```
*300L

0300-   A2 02       LDX   #$02
0302-   B5 00       LDA   $00,X
0304-   95 10       STA   $10,X
0306-   CA          DEX
0307-   8D 30 C0    STA   $C030
030A-   10 F6       BPL   $0302
030C-   00          BRK
030D-   00          BRK
030E-   00          BRK
030F-   00          BRK
0310-   00          BRK
0311-   00          BRK
0312-   00          BRK
0313-   00          BRK
0314-   00          BRK
0315-   00          BRK
0316-   00          BRK
0317-   00          BRK
0318-   00          BRK
0319-   00          BRK
*
```

OHJELMIEN KORJAILU

Kuten Lubarsky niin osuvasti totesi*: »There's always one more bug.» Ei kuitenkaan syytä huoleen, Monitori tarjoaa keinot kulkea ohjelmien läpi käsky käskyltä sen viimeisenkin virheen löytämiseksi. Monitorin STEP-käsky** tulkkaa, kirjoittaa ja toteuttaa käskyn kerrallaan, ja TRACE-käsky** kulkee nopeasti ohjelman läpi pysähtyen aina löytäessään BRK-käskyn.

Jokainen STEP-käsky saa Monitorin toteuttamaan sen käskyn, joka on ohjelmalaskurin näyttämässä osoitteessa. Käsky kirjoitetaan puretussa muodossa ja toteutetaan. Tämän jälkeen 6502:n sisäisten rekisterien arvot kirjoitetaan näkyviin ja ohjelmalaskuri siirretään osoittamaan ohjelman seuraavaa käskyä.

Kun kuljet STEP-käskyillä edellä syöttämäsi ohejlman läpi tapahtuu seuraavaa:

* Teoksesta Murphyn laki ja muita syitä, miksi asiat menevät pieleen.

** STEP- ja TRACE-komennot puuttuvat Appleista, joissa on Autostart ROM.

```
*3ØØS

Ø3ØØ-    A2 Ø2        LDX   #$Ø2
 A=ØA X=Ø2 Y=D8 P=3Ø S=F8
*S

Ø3Ø2-    B5 ØØ        LDA   $ØØ,X
 A=ØC X=Ø2 Y=D8 P=3Ø S=F8
*S

Ø3Ø4-    95 1Ø        STA   $1Ø,X
 A=ØC X=Ø2 Y=D8 P=3Ø S=F8
*12

ØØ12- ØC
*S

Ø3Ø6-    CA           DEX
 A=ØC X=Ø1 Y=D8 P=3Ø S=F8
*S

Ø3Ø7-    8D 3Ø CØ     STA   $CØ3Ø
 A=ØC X=Ø1 Y=D8 P=3Ø S=F8
*S

Ø3ØA-    1Ø F6        BPL   $Ø3Ø2
 A=ØC X=Ø1 Y=D8 P=3Ø S=F8
*S

Ø3Ø2-    B5 ØØ        LDA   $ØØ,X
 A=ØB X=Ø1 Y=D8 P=3Ø S=F8
*S

Ø3Ø4-    95 1Ø        STA   $1Ø,X
 A=ØB X=Ø1 Y=D8 P=3Ø S=F8
*
```

Huomaa, että kolmannen käskyn jälkeen tutkimme muistipaikan numero 12 sisällön. Se oli odottamamme, joten jatkoimme STEP-käskyillä. Monitor pitää ohjelmalaskurin ja viimeksi avatun osoitteen erossa toisistaan, joten muistin sisältöä voidaan lukea ja muuttaa samalla kun askelletaan ohjelman läpi.

TRACE-käsky on loputon askeltaja. Se lopettaa toimintansa vain kun painat RESETiä tai se tapaa BRK-käskyn ohjelmassa. Jos TRACE tapaa RTS-käskyllä monitoriin palaavan ohjelman lopun, se joutuu päättymättömään silmukkaan ja keskeytys on suoritettava RESETillä.

```
*T

Ø3Ø6-    CA           DEX
 A=ØB X=ØØ Y=D8 P=32 S=F8
Ø3Ø7-    8D 3Ø CØ     STA   $CØ3Ø
 A=ØB X=ØØ Y=D8 P=32 S=F8
Ø3ØA-    1Ø F6        BPL   $Ø3Ø2
```

```
 A=0B X=00 Y=D8 P=32 S=F8
0302-    B5 00       LDA   $00,X
 A=0A X=00 Y=D8 P=30 S=F8
0304-    95 10       STA   $10,X
 A=0A X=00 Y=D8 P=30 S=F8
0306-    CA          DEX
 A=0A X=FF Y=D8 P=B0 S=F8
0307-    8D 30 C0    STA   $C030
 A=0A X=FF Y=D8 P=B0 S=F8
030A-    10 F6       BPL   $0302
 A=0A X=FF Y=D8 P=B0 S=F8
030C-    00          BRK
030C-     A=0A X=FF Y=D8 P=B0 S=F8
*
```

REKISTERIEN TUTKIMINEN JA MUUTTAMINEN

Kuten edellä näit, STEP ja TRACE-käskyt kirjoittavat näkyviin 6502:n sisäisten rekisterien sisällön jokaisen käskyn jälkeen. Voit tutkia näiden rekisterien sisältöä milloin haulat tai asettaa ne ennakolta kun ajat konekielistä ohjelmaa STEP-, TRACE-, tai GO-käskyllä.

Monitor varaa viisi muistipaikkaa viidelle 6502:n rekisterille: A, X, Y, P (prosessorin tilan rekisteri) ja S (pinorekisteri). Monitorin EXAMINE-käsky – suoritetaan [CTRL E]:llä – käskee Monitoria kirjoittamaan näiden viiden rekisterin sisällön näkyviin ja siirtää 6502:n A-rekisteriä säilyttävän muistipaikan osoitteen seuraavaksi muutettavaksi osoitteeksi. Jos haluat muuttaa näiden muistipaikkojen arvoja, kirjoita kaksoispiste ja uuden arvot blankkojen erottamina. Seuraavalla kerralla kun annat Monitorille GO, STEP tai TRACE-käskyn, Monitor lataa nämä viisi arvoa asianmukaisiin rekistereihin ennen kuin toteuttaa ohjelman seuraavan käskyn.

```
*[CTRL E]

 A=0A X=FF Y=D8 P=B0 S=F8
*:B0 02

*[CTRL E]

 A=B0 X=02 Y=D8 P=B0 S=F8
*306S

0306-    CA          DEX
 A=B0 X=01 Y=D8 P=30 S=F8
*S

0307-    8D 30 C0    STA   $C030
 A=B0 X=01 Y=D8 P=30 S=F8
*S

030A-    10 F6       BPL   $0302
 A=B0 X=01 Y=D8 P=30 S=F8

*
```

SEKALAISIA MONITOR-KÄSKYJÄ

Voit valvoa COUT-alirutiinin (ks. sivu 32) käyttämää käänteinen/normaali-tilaa Monitorista niin, että kaikki Monitorin tulostus on käänteisenä. Syöttörivit ovat kuitenkin normaalitilassa. Monitorin tulostuksen saa palaamaan normaalitilaan NORMAL-käskyllä.

```
*Ø.F

ØØØØ- ØA ØB ØC ØD ØE ØF DØ Ø4
ØØØ8- C6 Ø1 FØ Ø8 CA DØ F6 A6
*I

*Ø.F

ØØØØ- ØA ØB ØC ØD ØE ØF DØ Ø4
ØØØ8- C6 Ø1 FØ Ø8 CA DØ F6 A6
*N

*Ø.F

ØØØØ- ØA ØB ØC ØD ØE ØF DØ Ø4
ØØØ8- C6 Ø1 FØ Ø8 CA DØ F6 A6
*
```

BASIC-käskyt — suoritetaan CTRL B:llä — antavat mahdollisuuden jättää Monitor ja siirtyä ROMiin varastoituun kieleen, tavallisesti joko Apple Integer BASICiin tai Applesoft II BASICiin. Kaikki entiset BASIC-ohjelmat ja -muuttujat häviävät. Jos olet siirtynyt BASICista Monitoriin ja haluat takaisin BASICiin ohjelma ja muuttujat ennallaan, käytä CTRL C (CONTINUE BASIC) -käskyä. Jos Apple Levynkäyttöjärjestelmä (DOS) on aktiivisena, käsky »3D0G» palauttaa sinut kieleen koskematta ohjelmaan tai muuttujiin.

PRINTER-käsky — joka tapahtuu CTRL P:llä — siirtää kaiken tavallisesti näyttöön tulevan tulostuksen Applen älykkäälle liitäntäkortille annettuun väylään Applen takaosassa. Väylän numeron tulisi olla väliltä 1—7, ja ko. väylässä olisi viisainta olla liitäntäkortti tai voit menettää Applen valvonnan ja ohjelmasi sekä muuttujat voivat hävitä. Käskyn muoto on:

{väylän numero} CTRL P

PRINTER-käsky väylään Ø kääntää tulostuksen takaisin Applen videonäyttöön.

KEYBOARD-käsky vaihtaa samalla tavalla Applen näppäimistön annettuun väylään liitettyyn oheislaitteeseen. Näiden käskyjen yksityiskohdista ja niiden BASIC-vastineista PR # :sta ja IN :sta löytyy tietoa suvulta 83 luvusta CSW- ja KSW-kytkimet. KEYBOARD-käskyn muoto on:

{väylän numero} CTRL K

Väylä numero Ø KEYBOARD-käskyssä saa Monitorin ottamaan oton Applen näppäimistöltä.

Monitor suorittaa myös yksinkertaista heksadesimaalista yhteen- ja vähennyslaskua. Kirjoita rivi muodossa:

{arvo} + {arvo}
{arvo} — {arvo}

Apple suorittaa laskut ja kirjoittaa tuloksen:

```
*2Ø+13
=33
*4A-C
=3E
*FF+4
=Ø3
*3-4
=FF
*
```

MONITORITRIKKEJÄ

Voit sijoittaa samalle riville niin monta Monitorikäskyä kuin haluat, kunhan vain erotat ne blankolla toisistaan ja merkkien kokonaismäärä on alle 254. Voit sekoittaa käskyjä täysin vapaasti, lukuunottamatta STORE (:) -käskyä. Koska Monitor ottaa kaksoispisteen perässä tulevat arvot ja sijoittaa ne peräkkäisiin muistipaikkoihin, viimeistä STOREn arvoa täytyy seurata kirjainkäsky ennen seuraavaa osoitetta. NORMAL-käsky on hyvä erotin: sillä ei tavallisesti ole vaikutusta ja sen voi sijoittaa mihin tahansa.

```
*3ØØ.3Ø7 3ØØ:18 69 1 N 3ØØ.3Ø2 3ØØS S

Ø3ØØ- ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
Ø3ØØ- 18 69 Ø1
Ø3ØØ-    18          CLC
 A=Ø4 X=Ø1 Y=D8 P=3Ø S=F8
Ø3Ø1-    69 Ø1       ADC    #$Ø1
 A=Ø5 X=Ø1 Y=D8 P=3Ø S=F8
*
```

Yksikirjaimisia käskyjä kuten L, S, I ja N ei tarvitse erottaa blankoilla.

Jos Monitor tapaa syöttörivillä merkin, jota se ei tunnista heksadesimaalimerkiksi eikä käskymerkiksi, se suorittaa kaikki käskyt ao. merkkiin asti, pysähtyy antaen äänimerkin ja jättää loppurivin huomiotta.

MOVE-käskyllä voidaan merkkijono moninkertaistaa jollekin muistialueelle.

Tämän tekemiseksi kirjoita ensin merkkijono alueen alkuun:

```
*300:11 22 33

*
```

Paina mieleesi merkkien lukumäärä, tässä tapauksessa kolme. Käytä sitten MOVE-käskyn erikoismuotoa:

{alku + lukumäärä} <{alku} . {loppu-lukumäärä} M

Tämä MOVE-käsky kirjoittaa ensin merkkijonon heti itsensä perään, sen jälkeen kirjoittaa tämän merkkijonon itsensä perään jne. kunnes se täyttää koko alueen.

```
*303<300.32DM

*300.32F

0300- 11 22 33 11 22 33 11 22
0308- 33 11 22 33 11 22 33 11
0310- 22 33 11 22 33 11 22 33
0318- 11 22 33 11 22 33 11 22
0320- 33 11 22 33 11 22 33 11
0328- 22 33 11 22 33 11 22 33
*
```

Samanlaista temppua voidaan käyttää VERIFY-käskylle kun halutaan nähdä esiintyykö joku merkkijono jatkuvasti samanlaisena kautta koko alueen.

```
*300:0

*301<300.31FM

*301<300.31FV

*304:02

*301<300.31FV

0303-00 (02)
0304-02 (00)
*
```

Voit myös luoda käskyrivin, joka toistaa itseään osittain tai kokonaan loputtomiin aloittamalla sen käskyrivin osan, joka tulee toistettavaksi kirjaimella, kuten N:llä, ja päättämällä sen merkeillä 34:n, missä n on heksadesimaalikuku, joka ilmoittaa merkin sijainnin silmukan alkavassa käskyssä. Rivin ensimmäiselle merkille n = 0. n:n arvoa täytyy seurata blankko, jotta silmukka toimisi kunnolla.

```
*N 300 302 34:0

0300- 11
```

```
Ø3Ø2- 33
Ø3ØØ- 11
Ø3Ø2- 33
Ø3ØØ- 11
Ø3Ø2- 33
Ø3ØØ- 11
Ø3Ø2- 33
Ø3ØØ- 11
Ø3Ø2- 33
Ø3ØØ- 11
Ø3Ø2- 33
Ø3Ø
*
```

Ainoa keino tällaisen silmukan pysäyttämiseksi on painaa RESETiä.

OMIEN KÄSKYJEN LUOMINEN

USER-käsky(CTRL Y), joka esiintyy syöttörivillä, pakottaa Monitorin hyppäämään osoitteeseen $3F8. Voit asettaa tähän osoitteeseen oman JMP-käskysi joka hyppää omaan ohjelmaasi. Ohjelmasi voi sitten joko tutkia Monitorin rekistereitä ja osoittimia tai itse syöttöriviä. Tässä esimerkiksi on ohjelma, joka saa CTRL Y :n toimimaan kommenttimerkkinä: Kaikki, mikä seuraa sitä syöttörivillä, hypätään yli.

```
*F666G

!300:LDY $34

Ø3ØØ-   A4 34      LDY   $34
! LDA 2ØØ,Y

Ø3Ø2-   B9 ØØ Ø2   LDA   $Ø2ØØ,Y
! JSR FDED

Ø3Ø5-   2Ø ED FD   JSR   $FDED
! INY

Ø3Ø8-   C8         INY
! CMP #$8D

Ø3Ø9-   C9 8D      CMP   #$8D
! BNE 3Ø2

Ø3ØB-   DØ F5      BNE   $Ø3Ø2
! JMP $FF69

Ø3ØD-   4C 69 FF   JMP   $FF69
!3F8:JMP $3ØØ

Ø3F8-   4C ØØ Ø3   JMP   $Ø3ØØ
```

```
!SFF69G

* CTRL Y THIS IS A TEST.
THIS IS A TEST.

*
```

MONITOR-KÄSKYJEN YHTEENVETO

Muistin tutkiminen

{osoite}	tutkii osoitteen arvon.
{os. 1}. {os. 2}	Kirjoittaa kaikkien osoitteiden {os1} ja {os2}välissä olevien osoitteiden arvot.
RETURN	Kirjoittaa viimeksi avattua osoitetta seuraavien kahdeksan osoitteen arvot.

Muistin sisällön muuttaminen

{osoite}: {arvo} {arvo}..	Varastoi arvot peräkkäisiin muistipaikkoihin aloittaen {osoite} :sta.
:{arvo} {arvo}...	Varastoi arvot aloittaen seuraavasta muutettavasta osoitteesta.

Siirtäminen ja vertailu

{kohde}<{alku} {loppu} M	Kopioi arvot muistialueelta {alku}{loppu} alueelle, joka alkaa {kohde} :sta.
{kohde}<{alku} {loppu} V	Vertaa muistialueen {alku} . {loppu} arvoja {kohde}:sta alkavaan alueeseen.

Lataaminen ja varastointi nauhalle

{alku} . {loppu} W	Kirjoittaa muistialueen {alku} . {loppu} arvot nauhalle 10 sekunnin otsikon jälkeen.
{alku} . {loppu} R	Lukee arvot nauhalta, varastoi ne muistialueelle, joka alkaa {alku} :sta ja loppuu {loppu} :uun. Kirjoittaa »ERR», jos virheitä ilmaantuu.

Ohjelmien ajaminen ja listaus

{osoite} G	Siirtää valvonnan konekieliselle ohjelmalle, joka alkaa {osoite} :sta.
{osoite} L	Purkaa ja näyttää 20 käskyä alkaen osoitteesta osoite . Seuraavat L:t näyttävät kukin 20 käskyä lisää.

Mini-Assembler

$ F666G	Käynnistää Mini-Assemblerin.*
$ {käsky}	Toteuttaa Monitor-käskyn Mini-Assemblerista.
$FF69G	Jättää Mini-Assemblerin.
{osoite} S	Purkaa, näyttää ja toteuttaa käskyn osoitteessa osoite ja näyttää 6502:n sisäisten rekistrien sisällön. Seuraavat S:t kirjoittavat ja toteuttavat seuraavat käskyt.
{osoite} T	Askeltaa loputtomiin. TRACE-käsky pysähtyy vain kun se kohtaa BRK-ohjeen tai RESET painetaan.
CTRL E	Kirjoittaa 6502:n sisäisten rekisterien sisällöt. **

Sekalaista

I	Asettaa päälle käänteisen tekstitilan.
N	Asettaa päälle normaalin tekstitilan.
CTRL B	Siirtyy Applen ROMiin varastoituun kieleen.
CTRL C	Palaa Applen ROMiin varastoituun kieleen.
{arvo} + {arvo}	Laskee yhteen kaksi arvoa ja kirjoittaa tuloksen.
{arvo} - {arvo}	Vähentää jälkimmäisen arvon ensimmäisestä ja kirjoittaa tuloksen.
{väylä} CTRL P	Siirtää annon laitteelle, jonka liitätäntäkortti on väylässä {väylä}. Jos{väylä}= 0, anto ohjataan Applen näytölle.
{väylä} CTRL K	Ottaa oton laitteelta, jonka liitäntäkortti on väylässä {väylä}. Jos{väylä}= 0, otto otetaan Applen näppäimistöltä.
CTRL Y	Hyppää konekieliseen alirutiiniin, joka alkaa osoitteesta $3F8.

* Ei käytettävissä Apple II Plus:ssa.

** Ei käytettävissä, jos Applessa on Autostart ROM.

JOITAKIN HYÖDYLLISIÄ MONITOR-ALIRUTIINEJA

Ohessa seuraa luettelo eräistä hyödyllisistä alirutiineista Apple Monitor ja Autostart ROMeissa. Jos aiot suorittaa näitä alirutiineja konekielisistä ohjelmista käsin, lataa oikeat muistipaikat tai 6502:n rekisterit, jotka alirutiini vaatii ja toteuta JSR-alituriinin alkuosoitteeseen. Se suorittaa toiminnon ja palaa takaisin. Tällöin 6502:n rekisterit ovat kuten seuraavassa esitetään.

$FDED COUT Tulosta merkki

Cout on tavallisin merkin tulostusalirutiini. Tulostettavan merkin tulee olla varastoituna akkuun. COUT kutsuu asianmukaista merkintulostusalirutiinia, jonka osoite on varastoitu CSW:hen (osoitteet$36 ja $37), tavallisesti COUT 1:tä (ks. alla).

$FDFO COUT 1 Tulosta näyttöön

COUT 1 tulostaa akussa olevan merkin Applen näyttöön tulostuskursorin senhetkiseen sijaintipaikkaan ja siirtää kursoria askeleen eteenpäin. Se ottaa huomioon normaali/käänteinen-muistipaikan. Se hoitaa kontrollimerkit RETURN, rivinsyöttö ja kello. Se palaa rekisterit koskemattomina.

$FE80 SETINV Aseta käänteinen tila

Asettaa käänteisen tilan COUT 1:lle. Kaikki tulostuvat merkit esitetään mustina pisteinä valkoisella pohjalla. Y-rekisteri asetetaan $3F:ksi, muut ennallaan.

$FE84 SETNORM Aseta normaalitila

Asettaa normaalin videotilan COUT 1:lle. Kaikki tulostuvat merkit esitetään valkeina pisteinä mustalla pohjalla. Asettaa Y-rekisterin $FF:ksi, muut ennallaan.

§FD8E CROUT Muodosta RETURN

CROUT lähettää RETURN-merkin käytössä olevaan tulostusvälineeseen.

$FD8B CROUT 1 RETURN ja clear

CROUT 1 tyhjentää näytön kursorin senhetkisestä paikasta oikeaan laitaan ja kutsuu sitten CROUTia.

$FDDA PRBYTE Kirjoita tavu hesadesimaalisena

Tämä alirutiini kirjoittaa akun sisällön heksadesimaalisena käytössä olevalle tuostusvälineelle. Akun sisältö menee sekaisin.

$FDE3 PRHEX Kirjoita heksadesimaalinumero

Tämä alirutiini kirjoittaa akun alemman puoliskon (nybble) yhtenä heksadesimaalimerkkinä. Akun sisältö menee sekaisin.

$F941 PRNTAX Kirjoita A ja X heksadesimaalisina

Tämä tulostaa A- ja X-rekisterien sisällöt yhtenä nelinumeroisena heksadesimaalilukuna. Akku sisältää ensimmäisen tulostustavun, X seuraavan. Akun sisältö menee yleensä sekaisin.

$F948 PRBLNK Kirjoita 3 blankkoa

Kirjoittaa kolme blankkoa käytössä olevalle tulostusvälineelle. Palattaessa akulla on tavallisesti arvo $A0, X-rekisterillä 0.

$F94A PRBL2 Kirjoita monta blankkoa

Tämä alirutiini tulostaa 1-256 blankkoa käytössä olevalle tulostusvälineelle. Alirutiiniin saavuttaessa X-rekisterin tulisi sisältää tulostettavien blankkojen lukumäärä. Jos X = $00, PRBL2 tulostaa 256 blankkoa.

$FF3A BELL Tulosta kello-merkki

Tämä alirutiini lähettää kellomerkin (CTRL G) käytössä olevalle tulostusvälineelle. Se jättää akkuun arvon $87.

$FBDD BELL 1 Äänimerkki Applen kaiuttimella

Tämä alirutiini päästää Applen kaiuttimella .1 sekuntia kestävän 1KHz:n äänen. Se sekoittaa A- ja X-rekisterit.

$FDOC RDKEY Hae syötetty merkki

Tämä on standardi-merkin ottoalirutiini. Se asettaa vilkkuvan syöttökursorin tulostuskursorin paikalle ja hyppää käytössä olevaan syöttöalirutiiniin, jonka osoite on varastoituna KSW:ssa (osoitteet$38 ja $39), tavallisesti KEYINiin (ks. alla).

$FD35 RDCHAR Hae syötetty merkki tai ESC-koodi

RDCHAR on vaihtoehtoinen syöttörutiini, mutta se myös tulkitsee kaikki yksitoista escapekoodia (ks. sivu 34).

$FD1B KEYIN Applen näppäimistön luku

Tämä on näppäimistöltäsyöttöalirutiini. Se lukee Applen näppäimistön, odottaa näppäimenpainallusta ja satunnaistaa satunnaislukugeneraattorin perusluvun (ks. sivu 32). Saatuaan näppäimenpainalluksen se poistaa vilkkuvan kursorin ja palaa painetun merkin koodi taltioituna akkuun.

$FD6A GETLN Hae syöttörivi promptin avulla

GETLN on alirutiini, joka kerää syöttörivit (ks. sivu 33). Ohjelma voi kutsua GETLNia, kun

oikea prompt on osoitteessa $33. GETLN toimittaa syöttörivin syöttöpuskuriin (joka alkaa osoitteesta $200) ja syöttörivin pituuden X-rekisteriin.

$FD67 GETLNZ Hae syöttörivi

GETLNZ on GETLNin vaihtoehtoinen alkukohta, joka lähettää vaunun palautuksen (RETURN) standarditulostukseen ennen siirtymistä GETLNiin.

$FD6F GETLN1 Hae syöttörivi, ei promptia

GETLN1 on vaihtoehtoinen GETLNin sisääntulokohta, joka ei kirjoita promptia ennen syöttörivin kokoamista. Jos käyttäjä kuitenkin peruuttaa syöttörivin joko liian monilla peruutusmerkeillä tai CTRL X:llä, GETLN1 käyttää muistipaikan $33 sisältöä promptina hakiessaan uuden rivin.

$FCA8 WAIT Odota

Tämä alirutiini odottaa tietyn aikaa ja palaa sitten sitä kutsuneeseen ohjelmaan. Odotusajan määrä riippuu akun sisällöstä. Kun A on akun sisältö, odotus on $1/2(26+27A+5A^2)$ sekuntia. WAIT nollaa akun ja jättää X- ja Y-rekisterit ennalleen.

$F864 SETCOL Aseta perusgrafiikan väri

Tämä alirutiini asettaa perusgrafiikkanäytöllä käytetyn värin tilalle akkuun tulleen värin. Sivulla 17 on taulukko perusgrafiikan väreistä.

$F85F NEXTCOL Kasvata väriä kolmella

Lisää 3 perusgrafiikan käyttämän värin numeroon.

$F800 PLOT Piirrä ruutu perusgrafiikkanäytölle

Tämä alirutiini piirtää perusgrafiikkanäytölle yhden ruudun ennaltamäärättyä väriä. Ruudun pystysuora sijainti on akussa ja vaakasuora sijainti Y-rekisterissä. PLOT sekoittaa akun, mutta jättää X- ja Y-rekisterit ennalleen.

$819 HLINE Piirrä vaakasuora ruuturivi

Tämä alirutiini piirtää perusgrafiikkanäytölle vaakasuoran rivin ennaltamäärätyn värisiä ruutuja. HLINEa pitää kutsua pystysuora koordinaatti akussa, vasemmanpuoleinen koordinaatti Y-rekisterissä ja oikeanpuoleinen koordinaatti muistipaikassa $2C. HLINE sekoittaa A:n ja Y:n muttei X:ää.

$F828 VLINE Piirrä pystysuora ruuturivi

Tämä alirutiini piirtää pystysuoran ruuturivin ennaltamäärättyä väriä perusgrafiikkanäytölle. VLINEa pitää kutsua vaakasuora koordinaatti Y-rekisterissä, ylin pystysuora koodinaatti akussa

ja lain pystysuora koordinaatti osoitteessa $2D. VLINE sekoittaa akun sisällön.

$F832 CLRSCR Tyhjennä koko perusgrafiikkanäyttö

CLRSCR tyhjentää koko perusgrafiikkanäytön. Jos sitä kutsuu, kun tekstitila on päällä, se täyttää koko taulun käänteisvärisillä »@ »-merkeillä. CLRSCR tuhoaa A:n ja Y:n sisällön.

$F836 CLRTOP Tyhjennä perusgrafiikkanäytön yläosa

CLRTOP tekee saman kuin edellinen, paitsi että se tyhjentää vain 40 ylintä riviä.

$F871 SCRN Lue perusgrafiikkanäyttö

Tämä alirutiini tuo näytön yksittäisen ruudun värin. Sitä kutsutaan samoin kuin PLOTia (ks. edellä). Värin numero palautetaan akkuun. Muihin rekistereihin ei kosketa.

$FB1E PREAD Lue peliohjain

PREAD lukee numeron, joka esittää peliohjaimen asentoa. Peliohjaimen numeron (0:sta 3:een) tulisi olla X-rekisterissä. Jos tämä numero ei ole oikea, tapahtuu kummia. PREAD toimittaa Y-rekisteriin numeron, joka on väliltä $00–$FF. Akun sisältö sekoitetaan.

$FF2D PRERR Kirjoita »ERR»

Lähettää sanan »ERR» kellomerkin seuraamana käytössä olevaan tulostusvälineeseen. Akun sisältö sekoitetaan.

$FF4A IOSAVE Kirjoita muistiin kaikki rekisterit

6502:n sisäisten rekisterien arvot kirjoitetaan muistiin osoitteisiin $45–$49 järjesyksessä A–X–Y–P–S. A:n ja X:n sisällöt muutetaan, desimaalitila poistetaan.

$FF3F IOREST Varastoi kaikki rekisterit

6502:n sisäisten rekisterien arvot sijoitetaan osoitteisiin $45–$49.

MONITORIN ERIKOISMUISTIPAIKAT

Taulukko 14: Sivun kolme Monitorimuistipaikat			
Osoite: Desim.	Hex	Käyttö: Monitori ROM	Autostart ROM
1008 1009	$3F0 $3F1	Ei käytössä.	Pitää sisällään muistipaikan osoitteen, joka käsittelee konekielisen keskeytyspyyntötoiminnon (BRK-request) (tavallisesti $FA59)
1010 1011	$3F2 $3F3	Ei käytössä.	Paluu (soft entry) vektori
1012	$3F4	Ei käytössä.	Käynnistystavu
1013 1014 1015	$3F5 $3F6 $3F7	Sisältää hyppykäskyn (JMP) aliohjelmaan, joka käsittelee Applesoft II:n »8»-käskyjä. Tavallisesti $4C $58 $FF.	
1016 1017 1018	$3F8 $3F9 $3FA	Sisältää hyppykäskyn (JMP) aliohjelmaan, joka käsittelee monitorin (CTRL Y) -käskyt.	
1019 1020 1021	$3FB $3FC $3FD	Sisältää hyppykäskyn (JMP) aliohjelmaan, joka käsittelee NMI-keskeytyspyynnöt (Nom Maskable Interrupts).	
1022 1023	$3FE $3FF	Sisältää osoitteen aliohjelmaan, joka käsittelee IRQ-keskeytyspyynnöt (Interrupt Requests).	

MINI-ASSEMBLER -KÄSKYJEN MUODOT

Applen Mini-Assembler tuntee 56 muistikasta (mnemonic) ja 13 osoitemuotoa, joita 6502:n assembler käyttää. Muistikkaat ovat standardeja, kuten esimerkiksi MOS Technology/Synertek 6500 Programming Manualissa (Apple numero A210003), mutta osoitemuodot eivät. Niinpä ne luetellaan tässä:

Taulukko 15: Mini-Assemblerin osoitusmuodot

Tapa:	Muoto:
Akku	Ei mahdollinen
Välitön ate	#$ (arvo)
Suora	$ (osoite)
Sivu nolla	$ (osoite)
Indeksoitu, sivu nolla	$ (osoite), X $ (osoite), Y
Indeksoitu, suora	$ (osoite), X $ (osoite), Y
Implied	Ei mahdollinen
Suhteellinen	$ (osoite)
Indeksoitu, epäsuora	($ (osoite), X)
Epäsuora, indeksoitu	($ (osoite), Y
Välitön, indeksoitu	($ (osoite))

Osoite käsittää yhden tai useampia heksadesimaalimerkkejä. Mini-Assembler tulkitsee osoitteet samoin kuin Monitorikin: jos osoitteessa on vähemmän kuin neljä merkkiä, se lisää nollia alkuun, jos niitä taas on enemmän kuin neljä, se ottaa neljä viimeistä.

Kaikki dollarimerkit ($), jotka osoittavat, että osoitteet ovat heksadesimaaleissa, hypätään yli ja voidaan jättää pois kirjoitettaessa.

Absoluuttisen ja Nollasivun osoitemuotojen välillä ei ole kieliopillista eroa. Jos Mini-Assemblerille annetaan käsky, joka voi olla kumpaa muotoa tahansa, Mini-Assembler tulkitsee käskyn olevaksi absoluuttisessa muodossa, jos käskyn operandi on suurempi kuin $FF, ja nollamuodossa, jos operandi on pienempi kuin $0100.

Accumulator ja Implied osoitusmuodot eivät tarvitse operandia.

Branch-osoitteet, jotka käyttävät relatiivista osoitusta, tarvitsevat maaliosoitteen. Mini-Assembler selvittää automaattisesti käskyssä tarvittavan suhteellisen etäisyyden. Jos maaliosoite on kauempana kuin 127 muistipaikan päässä käskystä, Mini-Assembler piipittää, kirjoittaa circumflexin (^) maaliosoitteen alle eikä reagoi riviin.

Jos Mini-Assemblerille annetaan muistikas ohjeeksi sekä operandi, eikä käsky voi käyttää annetun operandin osoitusmuotoa, Mini-Assembler ei hyväksy riviä.

LUKU 4
MUISTIN RAKENNE

Applen 6502 mikroprosessori voi viitata suoraan yhteensä 65.536 muistipaikkaan. Applen muistia voi kuvitella kirjana, jossa on 256 sivua ja joka sivulla 256 muistipaikkaa. Esimerkiksi »sivu $30» on 256 muistipaikan jakso, joka ulottuu osoitteesta $3000 osoitteeseen $30FF. Koska 6502 käyttää kahta kahdeksan bitin tavua muistipaikan osoitteena, voi toisen näistä tavuista kuvitella sivunumeroksi, ja toinen taas osoittaa muistipaikan sijainnin sivulla.

Applen 256-sivuinen muisti koostuu kolmesta osasta: keskusmuisti (RAM), järjestelmämuisti (ROM) ja otto/anto-toimintojen käyttämät muistipaikat (I/O). Muistin eri alueet on jaettu eri funktioiden kesken. Applen perusmuistikartta näyttää seuraavalta:

Järjestelmän muistikartta		
Sivun numero:		
Desim.	Hex	
0	$00	RAM (48K)
1	$01	
2	$02	
.	.	
.	.	
.	.	
190	$BE	
191	$BF	
192	$C0	I/O (2K)
193	$C1	
.	.	
.	.	
.	.	
198	$C6	
199	$C7	
200	$C8	I/O ROM (2K)
201	$C9	
.	.	
.	.	
.	.	
206	$CE	
207	$CF	
208	$D0	ROM (12K)
209	$D1	
.	.	
.	.	
.	.	
254	$FE	
255	$FF	

Kuvio 5: Järjestelmän muistikartta

RAM-MUISTI

Keskusmuistille varattu Applen muistikartan alue alkaa sivun 0 alusta ja ulottuu sivulle 191.

Applen emolevyn muistikapasiteetti ulottuu 4K:sta (4.096 tavusta) 48K:uun (49.152 tavuun) keskusmuistia. Lisäksi Applen keskusmuisti voidaan laajentaa 64K:uun (65.536 tavuun) lisäämällä Apple Language Card (Apple numero A2B0006). Tämä ylimääräinen 16K:n lukumuisti tulee Applen järjestelmämuistin tilalle siten, että kaksi 4K:n muistisegmenttiä jakaa 4K:n muistialueen, joka ulottuu $D000:sta $DFFF:ään.

Suurin osa Applen keskusmuistista on käytettävissä ohjelmien ja datan säilyttämiseen. Jotkin Applen keskusmuistialueet on kuitenkin varattu System Monitorille, eri kielille ja muille systeemitoiminnoille. Tässä seuraa kartta käytössä olevista keskusmuistialueista:

Taulukko 16: RAM-muistin järjestely ja käyttö

Sivun numero: Desim.	Hex	Käyttö:	
Ø	$ØØ	Systeemiohjelmat	
1	$Ø1	Pinorekisteri (stack)	
2	$Ø2	GETLN ottopuskuri	
3	$Ø3	Monitorin vektori	
4 5 6 7	$Ø4 $Ø5 $Ø6 $Ø7	Teksti ja perusgrafiikka Sivu 1	
8 9 1Ø 11	$Ø8 $Ø9 $ØA $ØB	Teksti ja perusgrafiikka Sivu 2	VAPAA RAM- MUISTI
12 through 31	$ØC $1F		
32 through 63	$2Ø $3F	Tarkkuusgrafiikka Sivu 1 Storage	
64 through 95	$4Ø $5F	Tarkkuusgrafiikka Sivu 2	
96 through 191	$6Ø $BF		

Seuraava katsaus osoittaa mitkä toiminnot varaavat mitenkin paljon muistialueita:

Sivu nolla: Apple 6502 mikroprosessorin rakenteen vuoksi alin sivu on konekileisten ohjelmien pääasiallinen käyttöalue. System Monitor käyttää noin 20 muistipaikkaa nollasivulta; Apple Integer BASIC käyttää muutaman lisää; Ja Applesoft II BASIC ja Applen levynkäyttöjärjestelmä (DOS) käyttävät loput. Taulut 18, 19, 20 ja 21 näyttävät näiden järjestelmätoimintojen käyttämät nollasivun muistipaikat.

Sivu yksi: Applen 6502 prosessori varaa kaikki sivun 1.256 tavua »pinoksi». Vaikka Apple tavallisesti käyttääkin alle puolet tästä sivusta kerrallaan, ei ole kovinkaan helppoa päätellä, mikä osa siitä on käytössä ja mikä lojuu vapaana, joten sivua 1 ei pitäisi käyttää tietojen varastoimiseen.

Sivu kaksi: GETLN-alirutiini, jota monet ohjelmat ja kielet käyttävät syöttörivien hakemiseen, käyttää sivua 2 syöttöpuskurina. Jos olet varma, ettet kirjoita pitkiä syöttörivejä, voit (jokseenkin) turvallisesti säilöä tietoa tilapäisesti sivun 2 ylempiin muistipaikkoihin.

Sivu kolme: Applen Monitor ROM (sekä Autostart että alkuperäinen) käyttävät ylimmät 16 muistipaikkaa sivulta kolme, numerosta $3F0 numeroon $3FF (desimaaleissa 1008–1023): Näistä muistipaikoista Monitorin käyttämät käsitellään sivulla 62.

Sivut neljästä seitsemään: Tämä 1.024 tavun muistialue on varattu Tekstin ja perusgrafiikan ensimmäiseksi sivuksi eikä sille siten voi varastoida dataa. Tällä alueella on 64 muistipaikkaa, joita ei esitetä näytössä. Ne on varattu oheislaitekorttien käyttöön (ks. sivu 82).

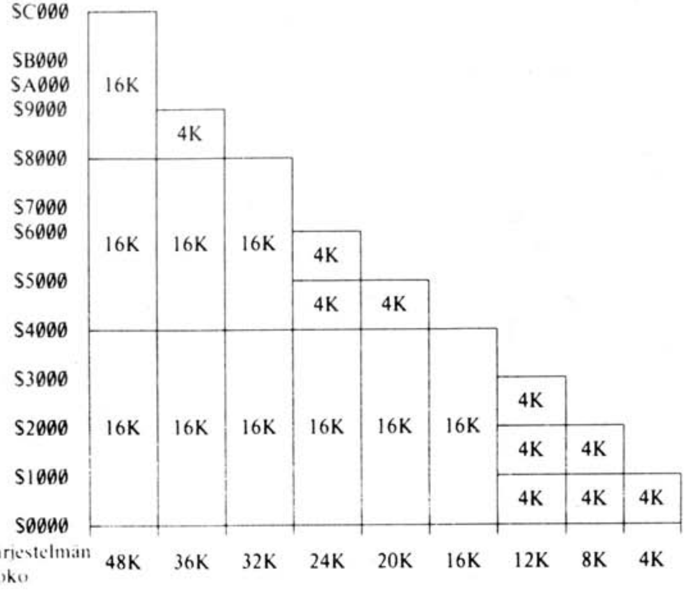
KESKUSMUISTIN OSAKOKONAISUUDET

Applen keskusmuisti koostuu 8–24 integroidusta piiristä. Nämä sijaitsevat kolmessa piirikantarivissä emolevyllä. Jokainen rivi voi sisältää kahdeksan muistipiiriä, joiden kapasiteetti puolestaan on joko 4.096 bittiä (4K) tai 16.384 bittiä (16K). 4K RAM-piirit kuuluvat Mostekin »4096»-perheeseen ja niissä voi olla merkintä MK4096 tai MCM6604. 16K piirit ovat »4116»-tyyppiä ja niissä on tunnus MK4116 tai UPD4160. Joka rivillä on oltava kahdeksan samantyyppistä piiriä, mutta eri rivit voivat sisältää keskenään erityyppisiä yksiköitä.

Kahdeksan 16K:n integroidun piirin rivi edustaa 16.384 kahdeksan bitin RAM-tavua. Rivin vasemmanpuoleisin integroitu piiri edustaa tavun alinta (vähiten merkitsevää) bittiä ja oikeanpuoleisin ylintä (merkitsevintä) bittiä. Se rivi, joka on lähinnä Applen etuosaa sisältää sen muistialueen, joka alkaa muistipaikasta 0. Seuraava rivi jatkuu siiitä mihin edellinen päättyi.

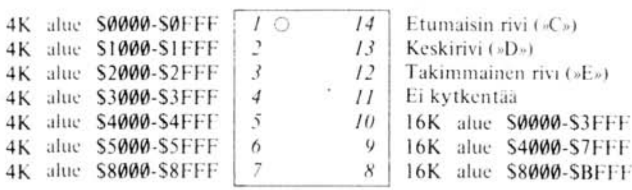
Applelle voi kertoa miten paljon muistia sillä on ja minkätyyppistä se on kytkemällä muistin määrittelyliittimet kolmeen integroitujen piirien kantaan emolevyn vasemmassa laidassa. Nämä liittimet ovat 14-jalkaisia IC-juotoskantoja, jotka näyttävät isoilta, laatikkomaisilta piireiltä. Mutta niissä ei ole integroituja piirejä; vain kolme yhdysjohtoa jokaisessa. Ne liittävät rivin RAM-piirit tiettyyn paikkaan Applen muistikartassa. Kaikki kolme liitintä tulisi yhdistää samalla tavalla. Apple toimittaa useita liittimiä tavallisimpiin systeemeihin. Valikoima näitä asennettiin Appleesi kun se rakennettiin ja saat uuden joukon niitä joka kerta kun osatat lisää muistia Appleen. Jos haluat kuitenkin laajentaa keskusmuistia muistipiireillä, joita et ole ostanut Appleltä, voit joutua suunnittelemaan omat liittimesi (tai modifioimaan vanhoja).

Yhdeksän erilaista muistielementtiyhdistelmää ovat mahdollisia. Nämä yhdeksän muistikokoa syntyvät erilaisista 4K:n ja 16K:n RAM-piirien yhdistelmistä. Ne ovat:



Kuvio 6. Muistirakenteet

Jokaisen määrittelyliittimen neljästätoista jalasta kolmen oikeassa yläkulmassa olevaa (ylhäältä päin katsottuna) vastaa kolmea RAM-piirin riviä Applen emolevyllä. Jokaisesta näistä juotostorneista pitäisi lähteä yhdysjohto toisen liittimen torniin. Tämä »toinen torni» vastaa sitä paikkaa Applen muistikartassa, johon kukin RAM-piiri-rivi halutaan liittää. Ko. liittimen tornit esitetään seuraavassa kuvassa:



Kuvio 7. Muistirakenteen valintaliittimen kytkentä

Jos rivillä on kahdeksan 16K:n piiriä, pitäisi riviä vastaava yhdysjohto liittää torniin, joka vastaa 16K:n muistialaa. Jos rivi taas sisältää 4K:n piirejä, pitäisi yhdysjohto liittää 4K:n muistitilaa vastaavaan juotostorniin. 4K piirien riviä ei koskaan saa yhdistää 16K:uun eikä kääntäen. Ei ole myöskään suotavaa jättää riviä yhdistämättä tai yhdistää kahta riviä samaan muistialueeseen.

On syytä varmistaa, että aina on käytössä jonkinlainen muisti, joka alkaa muistipaikasta 0. Applen muistin tulisi olla yksi jatkuva alue, mutta näin ei aina välttämättä ole. Jos esimerkiksi Sinulla on vain kolme 4K:n piirin riviä, mutta haluat käyttää tarkuuusgrafiikan ensimmäisen sivun, pitää Sinun yhdistää yksi 4K piiri-rivi muistin alkuun (alueelle $6000–$0FFF) ja kaksi

muuta riviä tarkkuusgrafiikan ensimmäisen sivun käyttämälle muistialueelle (muistipaikat $2000–$2FFF ja $3000–$3FFF) Näin saat käyttöösi 4 kilotavua keskusmuistia työalueeksi ja 8K RAMia kuvapuskuriksi.

Huomaa, että muistin määrittelyliittimet asennetaan etusivu (sivu, jolla on valkoinen piste) Applen etureunaan päin.

Apple Version 0 -levyt, joilla on 20 tai 24 K RAMia, tuottavat ongelmia. Näissä systeemeissä muistialue $4000–$5FFF kaksinkertaistuu alueella $6000–$7FFF riippumatta siitä, sisältääkö se RAMia vai ei. Niinpä systeemit, joilla on 20K tai 24K RAMia näyttävät siltä kuin niissä olisi 24K tai 36K RAMia, mutta ylimääräinen muisti on vain kuviteltua. Tämä piirre on poistettu Version 1 -levyistä.

LUKUMUISTI (ROM)

Normaalisti Applella on 2K (2.048 tavua) – 12K (12.288 tavua) lukumuistia emolevyllään. Järjestelmämuisti voi pitää sisällään System Monitorin, pari erilaista BASICia, erilaisia systeemi- ja apuohjelmia tai valmiita alirutiinipakkauksia kuten Apple Programmer's Aid 1 ROM.

Applen järjestelmämuisti varaa ylimmät 12K (48 sivua) muistikartasta alkaen osoitteesta $D000. Applen oikean toiminnan varmistamiseksi muistin ylimmissä osissa täytyy olla jonkinlaista ROMia. Kun Applen virta käännetään päälle, mikroprosessorilla täytyy olla jonkinlainen ohjelma suoritettavana. Se hakee tämän ohjelman osoitetta muistikartan ylimmistä osista. Applessa osoite on ROMissa ja se viittaa ohjelmaan, joka on myös ROMissa. Tämä ohjelma suorittaa aloitusrutiinit ja sallii Sinun ruveta käyttämään Applea (tarkemmin kappaleessa RESET-rutiini, sivu 36).

Tässä on kartta Applen lukumuistista ja siinä nykyisin säilytetyt ohjelmat ja ohjelmapakkkaukset:

<table>
<tr><th colspan="4">Taulukko 17: ROM-muistin rakenne ja käyttö</th></tr>
<tr><th colspan="2">Sivunumero</th><th colspan="2" rowspan="2">Käyttäjä:</th></tr>
<tr><th>Desim.</th><th>Hex</th></tr>
<tr><td>208</td><td>$D0</td><td rowspan="2">Programmer's Aid 1 ROM-piiri</td><td rowspan="10">Applesoft II BASIC</td></tr>
<tr><td>212</td><td>$D4</td></tr>
<tr><td>216</td><td>$D8</td><td rowspan="2"></td></tr>
<tr><td>220</td><td>$DC</td></tr>
<tr><td>224</td><td>$E0</td><td rowspan="5">INTEGER BASIC</td></tr>
<tr><td>228</td><td>$E4</td></tr>
<tr><td>232</td><td>$E8</td></tr>
<tr><td>236</td><td>$EC</td></tr>
<tr><td>240</td><td>$F0</td></tr>
<tr><td>244</td><td>$F4</td><td>Apuohjelmat</td></tr>
<tr><td>248</td><td>$F8</td><td rowspan="2">Monitori ROM</td><td rowspan="2">Autostart ROM</td></tr>
<tr><td>252</td><td>$FC</td></tr>
</table>

Kuusi 24-jalkaista IC-kantaa sisältävät Applen järjestelmämuistin integroidut piirit. Jokaisessa kannassa voi olla yksi 9316B-tyyppinen, 2.048 kahdeksanbittistä tavua sisältävä lukumuisti. Vasemmanpuoleisin ROM sisältää muistikartan ylimmät 2K ROMia; oikeanpuoleinsin taas sisältää muistialueen, joka alkaa osoitteesta $DO. Jos jossain tukilevyssä ei ole ROMia, vastaavalle alueelle kuuluvien muistipaikkojen arvoja on mahdoton arvata ennakolta.

Apple Firmware Card voi poistaa käytöstä osan tai kaiken Applen emolevyn ROM-muistista ja korvata sen omallaan. Kun Apple Firmware Card on sijoitettu paikalleen, voit poistaa Applen levyllä olevan ROMin kääntämällä levyllä olevan ohjauskytkimen ylä-asentoon ja painamalla RESET iä tai viittaamalla muistipaikkaan $C080 (desimaali 49.280 tai -16.256). Applen emolevyllä olevan muistin voi tehdä jälleen käyttökelpoiseksi kääntämällä controllerin kytkimen alasasentoon ja painamalla RESET tai viittamalla muistipaikkaan numero $C081 desimaali 49.281 tai -16.255). Tarkempia tietoja löytyy liitteestä A, Applesoft BASIC Programming Reference Manualista.

I/O-MUISTIPAIKAT

$4.096 muistipaikkaa (16 sivua) on varattu otto (input)- ja anto (output)-toiminnoille. Tämä 4K:n alue alkaa muistipaikasta $C000 (desimaali 49.152 tai -26.384) ja ulottuu muistipaikkaan numero $CFFF (desimaali 53.247 tai -12.289). Koska nämä toiminnot ovat jokseenkin mutkikkaita, on niille omistettu oma lukunsa (luku 5).

SIVUN NOLLA MUISTIKARTAT

Taulukko 18: Monitor-ohjelman nollasivun käyttö

Desim.		Ø	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15
	Hex	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
Ø	$ØØ																
16	$1Ø																
32	$2Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
48	$3Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
64	$4Ø	●	●	●	●	●	●	●	●	●	●					●	●
8Ø	$5Ø	●	●	●	●	●	●										
96	$6Ø																
112	$7Ø																
128	$8Ø																
144	$9Ø																
16Ø	$AØ																
176	$BØ																
192	$CØ																
2Ø8	$DØ																
224	$EØ																
24Ø	$FØ																

Taulukko 19: Applesoft II BASICin nollasivun käyttö

Desim.		Ø	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15
	Hex	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
Ø	$ØØ	●	●	●	●	●	●					●	●	●	●	●	●
16	$1Ø	●	●	●	●	●	●	●	●	●							
32	$2Ø																
48	$3Ø																
64	$4Ø																
8Ø	$5Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
96	$6Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
112	$7Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
128	$8Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
144	$9Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
16Ø	$AØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
176	$BØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
192	$CØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●		
2Ø8	$DØ	●	●	●	●	●	●			●	●	●	●	●	●	●	●
224	$EØ	●	●	●		●	●	●	●	●	●	●					
24Ø	$FØ	●	●	●	●	●	●	●	●	●							

Taulukko 20: Applen DOS 3.2:n nollasivun käyttö																	
Desim.		Ø	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15
	Hex	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
Ø	$ØØ																
16	$1Ø																
32	$2Ø							●	●			●	●	●	●	●	●
48	$3Ø						●	●	●	●	●					●	●
64	$4Ø	●	●	●	●	●	●	●	●	●		●	●	●	●		
8Ø	$5Ø																
96	$6Ø								●	●	●	●					●
112	$7Ø	●															
128	$8Ø																
144	$9Ø																
16Ø	$AØ																●
176	$BØ	●															
192	$CØ											●	●	●	●		
2Ø8	$DØ									●							
224	$EØ																
24Ø	$FØ																

Taulukko 21: Integer BASICin nollasivun käyttö																	
Desim.		Ø	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15
	Hex	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
Ø	$ØØ																
16	$1Ø																
32	$2Ø																
48	$3Ø																
64	$4Ø											●	●	●	●		
8Ø	$5Ø						●	●	●	●	●	●	●	●	●	●	●
96	$6Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
112	$7Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
128	$8Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
144	$9Ø	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
16Ø	$AØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
176	$BØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
192	$CØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
2Ø8	$DØ	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
224	$EØ																
24Ø	$FØ																

LUKU 5
INPUT/OUTPUT-JÄRJESTELMÄT

Applen otto- ja antotoiminnot voidaan jakaa kahteen pääryhmään: niihin, jotka suoritetaan Applen emolevyllä ja niihin jotka väyliin asetetut oheislaitteiden liitäntäkortit tekevät. Molemmantyyppiset toiminnot keskustelevat mikroprosessorin ja ohjelmiesi kanssa 4.096 muistipaikan välityksellä. Tässä luvussa kuvataan muistikarttaa ja eri otto- ja antotoimintoja. Niissä käytettävä kovo (hardware) esitellään seuraavassa luvussa.

SISÄÄNRAKENNETTU OTTO/ANTO (I/O)

Useimmat Applen normaaleista I/O-toiminnoista kuvaillaan lyhyesti kappaleessa 1, »Sinuiksi Applen kanssa».

Applen levyllä olevia I/O-toimintoja valvoo 128 muistipaikkaa, jotka sijaitsevat alueella $C000–$C07E (desimaaleina 49.152–49.279 tai -16.348 – -16.257). Kaksikymmentäseitsemän eri toimintoa jakaa nämä muistipaikat, mistä voi päätellä, että joillakin toiminnoilla on useampia kuin yksi muistipaikka. Joissakin tapauksissa yhdellä toiminnolla voi olla jopa 16 muistipaikkaa. Nämä 128 muistipaikkaa jaetaan viiteen tyyppiin: dataotot, tilamerkit (STROBE), ohjelmalliset kytkimet, vaihtokytkimet ja »lippu-sisääntulot» (flag inputs).

Dataotot. Ainoa Applen levyllä oleva dataotto on muistipaikka, jonka arvo esittää Applen näppäimistön tilaa. Sen korkein bitti on lippu-sisääntulon (ks. alempana) kaltainen. Alemmat seitsemän bittiä ovat viimeksi painetun näppäimen ASCII-koodi.

Lippu-sisääntulo. Useimmat sisäänrakennetut muistipaikat ovat yhden bitin lippuja. Nämä liput ilmenevät ao. muistipaikassa olevan tavun korkeimmassa bitissä. Niillä on vain kaksi arvoa: »päällä» ja »sammutettu». Mikä tahansa kieli voi tarkastaa helposti lipun tilan. Kehittynyt kieli voi käyttää PEEK- tai vastaavaa käskyä; jos tutkittu arvo on suurempi tai yhtä suuri kuin 128, lippu on päällä. Jos se on peinempi kuin 128, lippu on sammutettu. Konekieliset ohjelmat voivat ladata lippu-muistipaikan sisällön johonkin 6502 sisäiseen rekisteriin (tai käyttää BIT-käskyä) ja haarautua lipun tilan mukaan. BMI-käsky aiheuttaa haarautumisen, jos lippu on päällä, BPL-käsky taas jos lippu on sammutettu.

Yhden bitin otot (napin painallukset), kasetti-input, näppäimistön tilailmaisin (strobe) ja peliohjain ovat kaikki tätä tyyppiä.

Strobe annot. Utility strobea, clear keyboard strobea ja game controller strobea valvotaan muistipaikoilla. Jos ohjelma lukee jonkin näiden muistipaikan sisällön, siihen muistipaikkaan liittyvä toiminto aktivoituu. Utility stroben tapauksessa pelien I/O-liittimen nastan jännite laskee +5 voltista nollaan .98 mikrosekunnin ajaksi ja kasvaa sitten akaisin +5:een. Keyboard stroben ollessa kyseessä, näppäimistön lippu-sisääntulo sammutetaan. Game Controller stroben tapauksessa taas kaikki peliohjainten lippu-otot sammutetaan ja aikavakio aloitetaan uudestaan.

Ohjelmasi voi laukaista näppäimistön/peliohjainten strobet myös kirjoittamalla niitä vastaaviin muistipaikkoihin, mutta sitä on varottava utility stroben ollessa kyseessä. Tällöin nimittäin tuotetaan kaksi .98 mikrosekunnin mittaista pulssia, joilla on väliä noin 24.43 nanosekuntia. Syynä tähän on menetelmä, jolla 6502 lukee muistipaikkaan; ensin se lukee muistipaikan sisällön ja kirjoittaa sitten sen päälle.

Tämä kaksoispulssi jää huomiotta näppäimistön peliohjainten strobeista, mutta voi aiheuttaa ongelmia, jos se tapahtuu utility strobessa.

Toggel switches. Kaksi muuta strobe-antoa on yhdistetty »kiikkuihin» joilla on vain kaksi tilaa. Joka kerta kun niihin liittyvästä muistipaikasta luetaan, vaihtokytkin »heilahtaa» toiseen tilaan. Nämä vaihtokytkimet käyttävät kasettitulostusta ja kaiutinta. Ei ole mitään kätevää keinoa saada selville ko. kytkinten kulloinenkin tila. Kytkinten luonteesta johtuen niihin tulisi vain lukea eikä kirjoittaa (ks. strobe annot, edellä).

Ohjelmalliset kytkimet ovat kaksiasentoisia kytkimiä, joiden kumpaakin asemaa kontrolloi yksi muistipaikka. Jos kytkimen toista asentoa vastaavaan muistipaikkaan viitataan, kytkin kääntyy siihen suuntaan ja kääntäen. Kytkin asetetaan tiettyyn asentoon riippumatta sen alkuperäisestä asennosta, eikä kytkimen vallitsevaa asentoa voi saada selville. Ohjelmallisia kytkimiä valvoviin muistipaikkoihin voi huoletta kirjoittaa: kaksi pulssia ajaa saman asian kuin yksikin (ks. strobe outputs, edellä). Ulostuloja ja kaikkia videotilavalitsimia ohjataan ohjelmallisilla kytkimillä.

Erikoismuistipaikat, jotka valvovat sisäänrakennettuja input- ja output-toimintoja, on järjestetty seuraavasti:

Taulukko 22: Sisäänrakennetut I/O-muistipaikat

	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
$CØØØ	Näppäimistön data-otto															
$CØ1Ø	Näppäimistön stroben-nollaus															
$CØ2Ø	Kasettiliitännän ulostulon sysäys															
$CØ3Ø	Kaiuttimen sysäys															
$CØ4Ø	Utility-strobe															
$CØ5Ø	gr	tx	nomix	mix	pri	sec	lores	hires	anØ		an1		an2		an3	
$CØ6Ø	cin	pb1	pb2	pb3	gcØ	gc1	gc2	gc3	toisto $CØ6Ø-$CØ67							
$CØ7Ø	Peliohjaimen Strobe															

Peliohjainten Strobe:

gr	Aseta grafiikkatila	tx	Aseta tekstitila
nomix	Aseta teksti/tai grafiikka-tila (kokonäyttö)	mix	Sekoitettu teksti/grafiikkatila
pri	Näytä perussivu (sivu 1)	sec	Näytä varasivu (sivu 2)
lores	Näytä perusgrafiikka	hires	Näytä tarkkuusgrafiikka
an	Ulostulot	pb	Näppäinsisäänmenot (peliohjain)
gc	Peliohjainten sisäänmenot	cin	Kasettisisäänmeno

OHEISLAITE I/O

Applen emolevyn takaosassa on kahdeksan pitkää väylää eli oheisliitäntää. Seitsemään näistä väylistä voi liittää minkä tahansa useista juuri Applelle suunnitelluista oheislaiteliitännöistä. Jotta oheislaitekortit olisivat yksinkertaisempia ja monipuolisempia käyttää, Applen piirit ovat varanneet 280 muistipaikkaa jokaiselle seitsemästä väylästä. On myös kahden kilotavun yhteinen alue, jonka kaikki Applen oheislaitekortit jakavat.

Jokainen emolevyn väylä on numeroitu. Vasemmanpuoleisin väylä on Ø ja oikeanpuoleisin väylä 7. Väylä Ø on erikoinen: se on tarkoitettu RAM-, ROM-, tai oheisliitäntä-laajennukseen. Kaikilla muilla väylillä (1–7) on erityiset ohjausliitännät, jotka ovat aktiivisia vain yksi kerrallaan.

OHEISLIITÄNTÄKORTIN I/O -AVARUUS

Jokaiselle väylälle on annettu 16 muistipaikkaa alkaen osoitteesta $C080 yleiseen otto- ja antokäyttöön. Väylälle Ø nämä muistipaikat ovat alueella $080–$089, väylälle 1 alueella $090–$09F jne. Jokainen oheisliitäntäkortti voi käyttää näitä muistipaikkoja miten parhaaksi katsoo. Oheisliitäntäkortti voi päätellä, milloin sitä kutsutaan tarkkailemalla linjaa 41 (DEVICE SELECT) liitännän välityksellä. Kun tämän tornin jännite putoaa Ø:aan, mikroprosessorin kutsuma osoite on jossakin ao. kortin 16 tavun alueeella. Oheislaitekortti voi sitten katsoa neljää alinta osoiteriviä nähdäkseen mitä sen kuudestatoista osoitteesta kutsuttiin.

Taulukko 23: Oheislaitekorttien I/O-muistipaikat

	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
$CØ8Ø										Ø						
$CØ9Ø										1						
$CØAØ										2						
$CØBØ					Otto/anto kullekin väylänumerolle					3						
$CØCØ										4						
$CØDØ										5						
$CØEØ										6						
$CØFØ										7						

OHEISLAITEKORTIN ROM-AVARUUS

Jokaiselle oheislaitekortille on varattuna yksi 256 tavun sivu. Tätä sivua käytetään tavallisesti pitämään yllä 256 tavua ROMia tai ohjelmoitavaa ROMia (PROMia), joka sisältää oheislaitekortin ohjelmia tai alirutiineita. Tällä tavoin oheislaitekortit voivat olla »älykkäitä»; ne sisältävät ajamiseensa tarvittavan varusohejlman; käyttjän ei tarvitse ladata erillisiä ohjelmia niiden käyttöön.

Väylän käyttöön varatun sivun numero on $Cn, missä n on väylän numero. Väylä Ø:lla ei ole omaa sivua, joten useimpia oheisliitäntäkortteja ei voi asettaa siihen. Nastan 1 signaali (I/O SELECT) tulee aktiiviseksi (putoaa + 5 voltista nollaan) kun mikroprosessori viittaa väylälle viivatulla sivulla olevaan osoitteeseen. Oheislaitekortit voivat käyttää tätä signaalia kytkeäkseen ROMinsa käyttöön ja käyttää kahdeksan alinta osoiteriviä, jotta jokaiselle PROMin tavulle saadaan osoite.

Taulukko 24: Oheislaitekorttien PROM-muistipaikat

	$ØØ	$1Ø	$2Ø	$3Ø	$4Ø	$5Ø	$6Ø	$7Ø	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
$C1ØØ										1						
$C2ØØ										2						
$C3ØØ										3						
$C4ØØ	PROM-tila kullekin väylänumerolle									4						
$C5ØØ										5						
$C6ØØ										6						
$C7ØØ										7						

VIHJEITÄ I/O-OHJELMOINTIA VARTEN

Oheislaitekortin PROMissa olevien ohjelmien tulisi olla liikuteltavia. Se tarkoittaa, että niiden pitäisi olla toimivia riippumatta siitä, mihin kohtaan Applen muistikarttaa ne sijoitetaan. Niiden ei tulisi sisältää absoluuttisia viittauksia itseensä. Niiden pitäisi suorittaa kaikki hypyt (JMP) ehdollisina tai haarautuvina (BRANCH).

Oheislaitekortin PROMin voi tietenkin täyttää ohjelmilla, jotka eivät ole siirrettäviä, ja ainoa tällöin ilmenevä hankaluus on, että kortit tulevat riippuviksi väylistä. Jos Sinulla on puutetta tilasta oheislaitekorttien PROMissa, voit säästää monta tavua tekemällä alirutiinit väylästä riippuviksi.

Oheislaitekortin PROM-aliohjelman tulisi ensimmäiseksi tallentaa kaikkien 6502:n sisäisten rekisterien arvot. Apple Monitor ROMissa on alirutiini nimeltä IOSAVE, joka tekee juuri tämän. Se varastoi sisäisten rekisterien arvot muistipaikkoihin $45–$49 järjestyksessä A–X–Y–P–S. Tämä alirutiini alkaa muistipaikasta $FF4A. Sen pari, nimeltään IORESTORE, lataa kaikki sisäisten rekisterien arvot näistä musitipaikoista. Tätä alirutiinia, joka sijaitsee osoitteessa $FF3F, tulisi kutsua ennen kuin PROM-alirutiini päättyy.

Useimmat yhden merkin otot ja annot menevät 6502:n akkuun. Annon aikana näyttöön menevä merkki on akussa korkein bitti asetetuna. Oton tapahtuessa alirutiinin tulisi siirtää ko. merkki akkuun, myös korkein bitti asetettuna.

Oheislaitekortin PROMin ohjelma voi päätellä, missä väylässä se on suorittamalla seuraavat käskyt:

```
Ø3ØØ-   2Ø 4A FF   JSR   $FF4A
Ø3Ø3-   78         SEI
Ø3Ø4-   2Ø 58 FF   JSR   $FF58
Ø3Ø7-   BA         TSX
Ø3Ø8-   BD ØØ Ø1   LDA   $Ø1ØØ,X
Ø3ØB-   8D F8 Ø7   STA   $Ø7F8
Ø3ØE-   29 ØF      AND   #$ØF
Ø31Ø-   A8         TAY
```

Kun nämä on suoritettu, sen väylän numero, johon kortti on kytkettynä, on 6502:n Y-rekisterissä. Sen muoto on $0n, missä n on väylän numero. ROM-ohjelma voi edelleen käsitellä tätä arvoa siirtämällä sen neljä bittiä vasemmalle, jolloin saadaan $n0.

```
Ø312-   ØA          ASL
Ø313-   ØA          ASL
Ø314-   ØA          ASL
Ø315-   ØA          ASL
Ø316-   AA          TAX
```

Ohjelma voi käyttää tätä X-rekisterin numeroa 6502:n indeksoidun osoitemuodon kanssa viittaamaan kuuteentoista I/O-muistipaikkaan, jotka on varattu joka kortille. Esimerkiksi käsky

```
Ø317-   BD 8Ø CØ    LDA   $CØ8Ø,X
```

lataa 6502:n akkuun oheisliitäntäkortin käyttämän ensimmäisen I/O-muistipaikan sisällön. Osoite $C080 on perusosoite kaikkien kahdeksan oheislaitekortin käyttämälle ensimmäiselle muistipaikalle.Osoite $C081 on toisen I/O-muistipaikan osoite jne. Ohessa ovat kaikkien kuudentoista I/O-muistipaikan perusosoitteet jokaiselle väylälle.

Taulukko 25: I/O-muistipaikkojen perusosoitteet

Perus-osoite	Väylä Ø	1	2	3	4	5	6	7
$CØ8Ø	$CØ8Ø	$CØ9Ø	$CØAØ	$CØBØ	$CØCØ	$CØDØ	$CØEØ	$CØFØ
$CØ81	$CØ81	$CØ91	$CØA1	$CØB1	$CØC1	$CØD1	$CØE1	$CØF1
$CØ82	$CØ82	$CØ92	$CØA2	$CØB2	$CØC2	$CØD2	$CØE2	$CØF2
$CØ83	$CØ83	$CØ93	$CØA3	$CØB3	$CØC3	$CØD3	$CØE3	$CØF3
$CØ84	$CØ84	$CØ94	$CØA4	$CØB4	$CØC4	$CØD4	$CØE4	$CØF4
$CØ85	$CØ85	$CØ95	$CØA5	$CØB5	$CØC5	$CØD5	$CØE5	$CØF5
$CØ86	$CØ86	$CØ96	$CØA6	$CØB6	$CØC6	$CØD6	$CØE6	$CØF6
$CØ87	$CØ87	$CØ97	$CØA7	$CØB7	$CØC7	$CØD7	$CØE7	$CØF7
$CØ88	$CØ88	$CØ98	$CØA8	$CØB8	$CØC8	$CØD8	$CØE8	$CØF8
$CØ89	$CØ89	$CØ99	$CØA9	$CØB9	$CØC9	$CØD9	$CØE9	$CØF9
$CØ8A	$CØ8A	$CØ9A	$CØAA	$CØBA	$CØCA	$CØDA	$CØEA	$CØFA
$CØ8B	$CØ8B	$CØ9B	$CØAB	$CØBB	$CØCB	$CØDB	$CØEB	$CØFB
$CØ8C	$CØ8C	$CØ9C	$CØAC	$CØBC	$CØCC	$CØDC	$CØEC	$CØFC
$CØ8D	$CØ8D	$CØ9D	$CØAD	$CØBD	$CØCD	$CØDD	$CØED	$CØFD
$CØ8E	$CØ8E	$CØ9E	$CØAE	$CØBE	$CØCE	$CØDE	$CØEE	$CØFE
$CØ8F	$CØ8F	$CØ9F	$CØAF	$CØBF	$CØCF	$CØDF	$CØEF	$CØFF
	I/O-muistipaikat							

OHEISLAITTEILLE VARATUT MUISTIPAIKAT (SCRATCHPAD RAM)

Jokaiselle kahdeksasta väylästä on varattu kahdeksan muistipaikkaa Applen RAMista. Nämä 64 muistipaikkaa ovat itse asiassa sivut $04–$07 tekstin ja perusgrafiikan varaaman muistialueen sisällä. Näiden muistipaikkojen sisältöä ei kuitenkaan näytetä kuvaputkella, eivätkä tavalliset näytöllä suoritettavat operaatiot vaikuta niihin*. Oheislaitekortit käyttävät näitä muistipaikkoja tietojen säilytykseen sillä aikaa, kun kortit ovat toiminnassa. Näillä »scratchpad»-muistipaikoilla on seuraavat osoitteet:

* Ks. teksti-ikkuna, sivu 31.

Taulukko 26: I/O »Scratcpad»-alueen RAM-osoitteet

Perus-osoite	Väylänumero 1	2	3	4	5	6	7
$Ø478	$Ø479	$Ø47A	$Ø47B	$Ø47C	$Ø47D	$Ø47E	$Ø47F
$Ø4F8	$Ø4F9	$Ø4FA	$Ø4FB	$Ø4FC	$Ø4FD	$Ø4FE	$Ø4FF
$Ø578	$Ø579	$Ø57A	$Ø57B	$Ø57C	$Ø57D	$Ø57E	$Ø57F
$Ø5F8	$Ø5F9	$Ø5FA	$Ø5FB	$Ø5FC	$Ø5FD	$Ø5FE	$Ø5FF
$Ø678	$Ø679	$Ø67A	$Ø67B	$Ø67C	$Ø67D	$Ø67E	$Ø67F
$Ø6F8	$Ø6F9	$Ø6FA	$Ø6FB	$Ø6FC	$Ø6FD	$Ø6FE	$Ø6FF
$Ø778	$Ø779	$Ø77A	$Ø77B	$Ø77C	$Ø77D	$Ø77E	$Ø77F
$Ø7F8	$Ø7F9	$Ø7FA	$Ø7FB	$Ø7FC	$Ø7FD	$Ø7FE	$Ø7FF

Väylä Ø:lla ei ole scratchpad RAM-osoitteita varattuna. Apple DOS 3.2 käyttää musitipaikkojen perusosoitteita ja kaikki oheislaitekortit jakavat ne. Jotkin näistä muistipaikoista on omistettu tietyille tehtäville; muistipaikka $7F8 säilyttää aktiivisena olevan kortin väylän numeron (muodossa $Cn) ja muistipaikka $5F8 säilyttää sen väylän numeroa, josta olevasta ohjainkortista mahdollinen aktiivinen DOS ladattiin.

Käyttämällä väylän numeroa $0n, joka on saatu edellä olevalla ohjelmalla, alirutiini voi viitata suoraan mihin tahansa kahdeksasta scratchpad-muistipaikastaan:

```
Ø31A-   B9 78 Ø4    LDA   $Ø478,Y
Ø31D-   99 F8 Ø4    STA   $Ø4F8,Y
Ø32Ø-   B9 78 Ø5    LDA   $Ø578,Y
Ø323-   99 F8 Ø5    STA   $Ø5F8,Y
Ø326-   B9 78 Ø6    LDA   $Ø678,Y
Ø329-   99 F8 Ø6    STA   $Ø6F8,Y
Ø32C-   B9 78 Ø7    LDA   $Ø778,Y
Ø32F-   99 F8 Ø7    STA   $Ø7F8,Y
```

CSW/KSW-KYTKIMET

Muistipaikkaparia $36 ja $37 (desimaali 54 ja 55) kutsutaan CSW:ksi – Character output Switch. Yksinään muistipaikkaa §36 kutsutaan CSWL:ksi (CSW Low) ja muistipaikkaa $37 CSWH:ksi (CSW High). Tämä muistipaikkapari sisältää sen alirutiinin osoitteen, jota Apple parhaillaan käyttää yhden merkin antoon. Tämä osoite on normaalisti $FDF0, COUT-alirutiinin osoite (ks. sivu 30). Monitorin PRINTER-käsky (CTRL P) ja BASICin PR -käsky voivat muuttaa tätä osoitetta oheislaitekortin PROMissa olevan alirutiinin osoitteeksi. Nämä molemmat käskyt asettavat osoitteen $Cn00 tähän muistipaikkapariin, missä n on annetun väylän numero. Apple kutsuu siten tätä alirutiinia joka kerta, kun sen on tulostettava merkki. Tämä alirutiini voi käyttää edellä annettuja käskyjä löytääkseen väylänsä ja käyttää I/O RAM ja scratchpad-osoitteita. Kun se on päässyt loppuun, se voi joko suorittaa RTS-käskyn (paluu alirutiinista) palatakseen ohjelmaan tai kieleen jossa se oli, tai hypätä COUT-alirutiiniin muistipaikkaan $FDF0, kirjoittaakseen merkin näyttöön ja palatakseen sitten antoa tuottaneeseen ohjelmaan.

Vastaavasti muistipaikkoja $38 ja $39 (desimaali 56 ja 57) kutsutaan erikseen KSWL:ksi ja

KSWH:ksi tai yhdessä KSW:ksi (Keyboard input Switch). Ne säilyttävät sen muistipaikan osoitteen, jossa olevalla alirutiinilla Apple parhaillaan suorittaa yhden merkin ottoa. Tämä osoite on tavallisesti $FD18, KEYIN-alirutiinin osoite. Monitorin KEYBOARD-komento (CTRL K) ja BASIC-käsky IN molemmat sijoittavat tähän osoitteeseen $Cn00, jossa n on jälleen annetun väylän numero. Apple kutsuu alirutiinia oheislaitekortin PROMilla aina kun se haluaa yhden merkin ko. antolaitteelta. Alirutiinin pitäisi sijoittaa syötetty merkki 6502:n akkuun ja palata alirutiiniin (RTS:llä). Alirutiinin pitäisi asettaa ylin bitti merkistä ennen kuin se palautetaan.

Oheislaitekortin PROMin alirutiinit voivat muuttaa CSW- ja KSW-kytkinten osoitteita osoittamaan jonnekin muualle PROMiin kuin aivan alkuun. Esimerkiksi tietty PROM voi alkaa koodisegmentillä, jotta saataisiin selville, mikä väylä on ajossa ja aloitusrutiinien hoitamiseksi ja hypätä sitten varsinaiseen merkintäkäsittelyyn. Aloitusrutiinin osana se voi muuttaa kulloin mahdollista KSW:n ja CSW:n osoittamaan suoraan merkintäkäsittelyrutiinia. Kun Apple sitten seuraavan kerran kysyy ottoa tai antoa ao. kortilta, käsittelyrutiinit hyppäävät jo kertaalleen tehdyn aloitusrutiinin yli ja käyvät käsiksi suoraan tehtävään. Tämä säästää aikaa nopeutta vaativissa tilanteissa.

Oheislaitekorttia voidaan käyttää sekä ottoon että antoon jos sen PROMissa on erilliset alirutiinit ja toiminnot ja vastaavasti CSW- ja KSW-kytkinten muutokset. Oheislaitekortin aloitusrutiini voi päätellä kaivataanko sitä ottoon vai antoon katsomalla CSW- ja KSW-kytkinten ylempiä osia. Se kytkin, jolla on arvo $Cn on juuri kutsumassa tätä korttia saadakseen toimintonsa suoritetuksi. Jos molemmat kytkimet sisältävät $Cn:n, alirutiinin tulisi otaksua, että sitä kaivataan antoa varten.

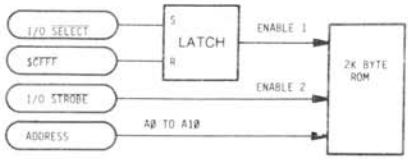
ROM-MUISTIN LAAJENTAMINEN

Kahden kilon muistialue muistipaikasta $C800 muistipaikkaan $CFFF on varattu oheislaitekortin 2K ROMille tai PROMille, säilyttämään ohjelmia tai ajamaan alirutiineja. ROMin laajennuksella on myös se etu, että se on absoluuttisesti sijoitettu Applen muistikarttaan, minkä vuoksi liitäntäohjelmien kirjoittaminen sinne on huomattavasti vapaampaa.

Tämä PROM-avaruus on kaikkien väylien käytössä ja useammalla kuin yhdellä kortilla voi olla ROMin laajennus. Kuitenkin vain yksi laajennettu ROM kerrallaan voi olla aktiivinen.

Jokaisella oheislaitekortin ROM-laajennuksella tulisi olla on/off-kytkin, joka tekee sen käytön mahdolliseksi. Tämän kytkimen pitäisi olla käännettävissä päälle $\overline{\text{DEVICE SELECT}}$ -signaalilla (samalla, joka salli 256-tavun PROMin käytön). Tämä tarkoittaa sitä, että ROMin laajennus on käytettävissä heti kun olet ensimmäisen kerran ilmoittanut kortille, että se on päällä. Toinen ROM-laajennuksen käytön mahdollistaja tulisi olla $\overline{\text{I/O-STROBE}}$-linja, jokaisen oheislaiteliitännän nasta 20. Tämä linja tulee aktiiviseksi kun Applen mikroprosessori viittaa laajennus-ROMin alueella olevaan muistipaikkaan. Kun tämä linja tulee aktiiviseksi ja aiemmin mainittu kytkin on päällä, Apple käyttää varsinaista laajennus-ROMia (ks. kuvio 8).

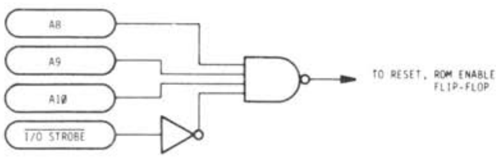
Oheislaitekortin 256 tavun PROM voi päästä laajennus-ROM-avaruuteen vain viittaamalla osoitteeseen $CFFF aloitusrutiinissaan. Tämä osoite on erikoisosoite ja kaikki oheisliitäntäkortit tuntevat sen pois-signaaliksi kytkimelleen ja menettävät oikeuden käyttää laajennus-ROMiaan. Tänä vie tietysti laajennus-ROMin myös kortilta, joka yrittää saada lisää muistiavaruutta, mutta ROM vapautuu jälleen kun mikroprosessori saa uuden käskyn 256 tavun PROMilta. Tällöin ROM on käytettävissä ja muistiavaruus on tyhjä. Alirutiinit voivat hypätä suoraan ROMissa oleviin ohjelmiin, jossa ne voivat käyttää 2K esteetöntä, absoluuttisesti numeroitua muistiavaruutta:



Kuvio 8: Laajennus ROM:in aktivointikytkentä

On mahdollista säästää piirikustannuksissa (ROMin kustannuksella) jättämällä erikoisosoitteen $CFFF dekoodaamatta kokonaan. Itse asiassa, jos Sinulla on varaa menettää viimeiset 256 tavua ROMia, seuraava piiri riittää hyvin:

```
0332-   2C FF CF    BIT   $CFFF
0335-   4C 00 C8    JMP   $C800
```



Kuvio 9: $CFXX-dekoodaus

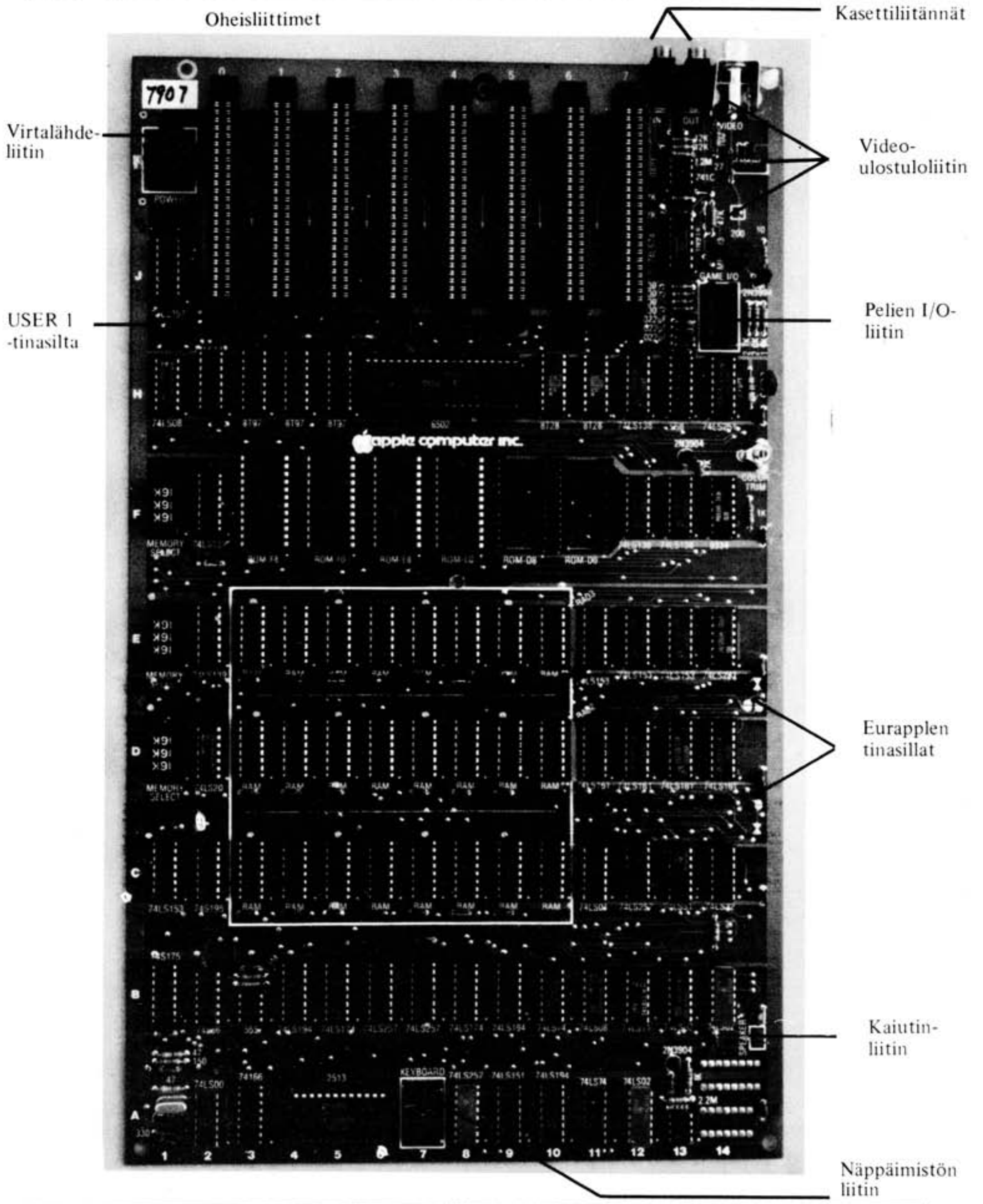
LUKU 6
LAITTEISTON RAKENNE

MIKROPROSESSORI

Mikroprosessori 6502

Malli:	MCS 6502/SY 6502
Valmistaja:	Mos Technology, Inc Synertek Rockwell
Käskyjen määrä:	56
Osoitusmuotoja:	13
Akkuja	1 (A)
Indeksirekisterejä:	2 (X, Y)
Muita rekisterejä:	Pino-osoitin (S) Prosessorin tilarekisteri (P)
Pino:	256 tavua, kiinteä
Tilaliput:	N (etumerkki) C (carry) V (ylivuoto)
Muita lippuja:	I (keskeytyskielto) D (desimaaliaritmetiikka) B (break)
Keskeytykset:	2 (IRQ, NMI)
Nollaukset (Reset):	1 (RES)
Osoiteavaruus:	2^{16} (64K) muistipaikkaa
Osoiteväylä:	16 bittiä, rinnakkainen
Dataväylä:	8 bittiä, rinnakkainen, kaksisuuntainen
Käyttöjännite:	+ 5 volttia
Tehohäviö	**.25 W**
Kellotaajuus:	1.023 MHz

Mikroprosessori saa pääkellotaajuutensa, Φ0 ja Φ1, edellä kuvatuista ajastuspiireistä. Signaalit ovat vastavaiheisia 1.023 MHz kellosignaaleja. Useat käsikirjat, kuten MOS Technology Hardware Manual, käyttävät määritelmää Φ 2 Applen Φ0 -kellolle.



Kuva 10. Applen pääkytkentälevy

Mikroprosessori käyttää osoite- ja dataväyliä vain kun Φ0 on aktiivinen. Kun Φ0 on epäaktiivinen, mikroprosessori suorittaa sisäisiä operaatioita eikä tarvitse data- ja osoitelinjoja.

Mikroprosessorilla on 16 bitin osoiteväylä ja 8 bitin kaksisuuntainen dataväylä. Osoiteväylän rivit on puskuroitu kolmeen 8T97 kolmitilaiseen puskuriin, jotka sijaitsevat levyllä kohdissa H3, H4 ja H5. Osoiterivit ovat auki vain DMA-jakson aikana ja aktiivisia muulloin. Osoiteväylän osoite tulee käyttökelpoiseksi n. 300 ns sen jälkeen, kun Φ0 on ollut aktiivinen ja pysyy sellaisena koko Φ0:n ajan.

Dataväylä on puskuroitu kahteen kaksisuuntaiseenkolmitilaiseen 8T28 -puskuriin, jotka sijaitsevat levyllä kohdissa H10 ja H11. Mikroprosessorista tuleva data-tieto sijoitetaan väylään n. 300 ns sen jälkeen, kun Φ1 ja READ/WRITE-signaali (R/W) ovat pudonneet nollaan. Muulloin mikroprosessori joko kuuntelee data-väylää tai ei reagoi siihen mitenkään.

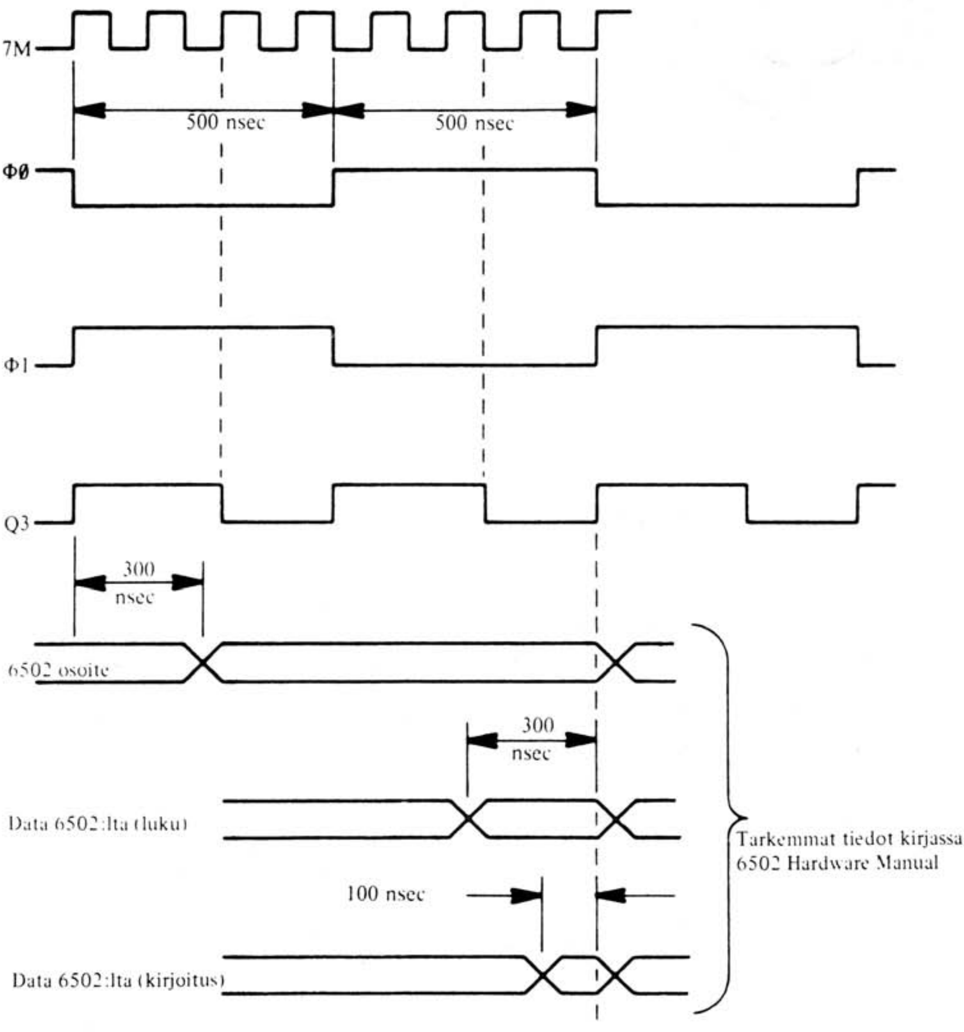
Mikroprosessoriin menevät RDY-, RES-, IRQ- ja NMI-linjat pidetään auki + 5 voltissa 3.3 ohmin resistorilla. Myös oheislaiteliitännät käyttävät näitä linjoja (ks. sivu 105)

SET OVERFLOW (SO) (ylivuodon asetus) -linja mikroprosessoriin on yhdistetty pysyvästi maahan.

JÄRJESTELMÄN AJASTUS

Taulukko 27: Ajoitussignaalien kuvaukset	
14M:	Perusoskillaattorin ulostulo, 14.318 MHz. Kaikki ajastussignaalit on jaettu tästä oskillaattoritaajuudesta.
7M:	Sisäinen ajastussignaali 7.159 MHz.
COLOR REF:	Värien referenssitaajuus, 3.580 MHz, videogeneraattorin käytettäväksi.
Φ0 (Φ2) :	Järjestelmän kellotaajuuden vaihe 0, 1.023 MHz, vastavaiheessa taajuuteen Φ1.
Φ1:	Järjestelmän kellotaajuuden vaihe 1 1.023 MHz, vastavaiheessa taajuuteen 0.
Q3:	Yleinen ajastussignaali, kaksi kertaa järjestelmän kellotaajuus, mutta epäsymmetrinen.

Kaikki oheislaiteliitännät saavat ajastussignaalit 7M, Φ0, Φ 1 ja Q3. Ajastussignaalit 14M ja COLOR REF eivät ole oheislaiteliitäntöjen käytössä.



Kuvio 11. Ajoitussignaalit ja niiden suhteet

VIRTALÄHDE

Applen virtalähde (U. S. Patentti #4,130,862)

Sisäänmenojännite:	107 VAC – 132 VAC, tai 214 VAC – 264 VAC (valittavissa kytkimellä)*
Ulostulojännitteet:	+5.0 +11.8 –12.0 –5.2
Tehonkulutus:	60 W maksimi (täysi kuorma) 79 W maksimi (hetkellinen**)
Maksimi ulostulovirrat:	+5V/2,5A –5V/250mA +12V/1,5A (2,5A hetkellinen**) –12V/250mA
Käyttölämpötila:	55 °C (131 °F)

Applen virtalähde on korkeajännitteinen hakkurityyppinen virtalähde. Kun useimmat muut virtalähteet käyttävät isoa muuntajaa, jossa on paljon käämejä, joilla syöttöjännite voidaan muuttaa useiksi pienemmiksi jännitteiksi, joita sitten tasasuunnataan ja säädetään, Applen virtalähde muuntaa ensin vaihtojännitteen tasajännitteeksi ja käyttää tällä tasajännitteellä sitten korkeataajuista oskillaattoria. Oskillaattorin tuotos syötetään pieneen muuntajaan, jossa on useita käämejä. Toisiokäämien jännitteistä saadaan sitten säätämällä lopulliset lähtöjännitteet.

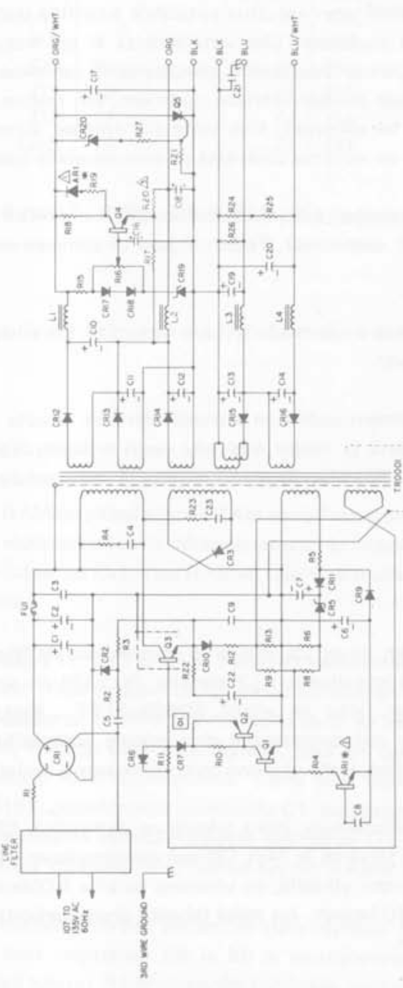
+ 5 voltin lähtöjännitettä verrataan sitten vakioon ja ero syötetään takaisin oskillaattoriin. Kun virtalähteen lähtöjännite alkaa siirtyä toleranssien ulkopuolelle, oskillaattorin taajuutta muutetaan ja jännite palaa normaalille tasolle.

Jos joku virtalähteen tulosjännitteistä sattuu olemaan oikosulussa, virtalähteen palautepiiri pysäyttää oskillaattorin ja katkaisee kaikki ulostulopiirit. Sitten virtalähde pitää noin puolen sekunnin tauon ja yrittää käynnistää oskillaation uudelleen. Jos ulostulo on yhä oikosuljettu, se pitää uuden tauon. Tätä jatkuu, kunnes oikosulku poistuu tai virta katkaistaan.

Jos virtalähteen ulostuloliitäntää ei ole yhdistetty Applen äitilevyyn, virtalähde huomaa tämän »ei-kuormitusta»-tilanteen ja oikosulkee itsensä. Tämä aktivoi edellä kuvatut suojapiirit ja katkaiseen kaiken virrantuotannon. Tällä tavoin virtalähteen sisäosia varjellaan vahingoilta.

* Jannitteenvalitsinta ei ole kaikissa Appleissa.

** Virtalahde voi toimia vahingoittumatta 20 minuuttia ajoittaisella ylikuormituksella, jos sitä seuraa 10 minuuttia normaalikuormitusta.



Kuva 12. Virtalähteen kytkentäkaavio

Jos joku ulostulojännitteistä ylittää toleranssirajansa, minkä tahansa virtalähteen sisäisen tai ulkoisen syyn akia, se sulkee jälleen itsensä estääkseen levyllä olevia komponentteja vahingoittumasta. Tämä takaa sen, että kaikki jännitteet ovat joka suhteessa oikeita tai ne katkaistaan.

Kun joku edellä mainituista virhetilanteista ilmenee, sisäiset suojapiirit pysäyttävät muuntajaa käyttävän oskillaattorin. Vähän ajan kuluttua virtalähde suorittaa uudellenaloitusjakson ja yrittää käynnistää oskillaatorin uudestaan. Jos virtalähdettä ei ole korjattu, virtalähde suljetaan uudestaan. Tämä jakso voi jatkua loputtomiin aiheuttamatta vahinkoa virtalähteelle. Joka kerta kun oskillaattori sammutetaan ja käynnistetään uudelleen, sen taajuus tulee kuultavalle alueelle ja sen voi kuulla vinkuvan tai vikisevän. Kun virhe siis ilmenee, virtalähteestä kuuluu tasainen »klik klik klik» -ääni. Tämä on varoitus siitä, että jotakin on vialla jossakin ulostulojännitteessä.

Missään tapauksessa ei muuntajaan tulisi syöttää enempää kuin 140 VAC (tai yli 280 VAC, jos virtalähteen kytkin on 220V -asennossa). Tämän ohjeen rikkominen voi vaurioittaa virtalähdettä pysyvästi.

Applen virtalähde tulee kytkeä suojamaadoitettuun liitäntään. On erittäin tärkeää, että Apple on yhdistetty kunnon maajohtoon.

VAROITUS: Applen virtalähteen sisällä on korkeajännitteitä. Suurta osaa sisäisistä piireistä EI ole eristetty verkkojännitteistä ja niiden käsittely vaatii erikoistyökaluja. ÄLÄ YRITÄ KORJATA VIRTALÄHDETTÄ ITSE! Vian ilmetessä toimita se valtuutetulle Apple-myyjälle.

ROM-MUISTI

Applessa voi olla korkeintaan kuusi 2K x 8 bitin ohjelmoitua ROM-piiriä. Yksi näistä otetaan käyttöön Applen levyn muistipaikassa F12 olevalla 74LS138:lla aina kun mikroprosessorin osoiteväylä sisältää osoitteen, joka on välillä $D000–$FFF. Kaikkien ROMien kahdeksan data-ulostuloa on yhdistetty mikroprosessorin data-linjojen puskureihin. Ja ROMin osoitelinjat on yhdistetty niihin puskureihin, jotka ohjaavat mikroprosessorin osoitelinjoja A0–A10.

ROMeilla on kolme piirin valintalinjaa, jotka tekevät ne aktiivisiksi. CS1 ja CS3, molemmat aktiivisia alhaalla, on kytketty toisiinsa ja 74SL138:aan muistipaikassa F12, joka valitsee yksityiset ROMit. CS2, joka on aktiivinen ylhäällä, on yhteinen kaikille ROMeille ja on yhdistetty oheislaitteiden INH (ROM Inhibit)-linjaan. Jos mikä tahansa oheislaitekortti vetää tämän linjan alas, kaikki levyn ROMit suljetaan.

ROMit ovat samanlaisia kuin tyyppien 2316 ja 2716 ohjelmoitavat ROMit. Kuitenkin useimmat näiden PROMien piirin valinnan ovat eri polariteettia eikä niitä voi kytkeä suoraan Applen levyyn.

A7	*1* ○	*24*	+5v
A6	*2*	*23*	A8
A5	*3*	*22*	A9
A4	*4*	*21*	$\overline{CS3}$
A3	*5*	*20*	$\overline{CS1}$
A2	*6*	*19*	A1Ø
A1	*7*	*18*	CS2
AØ	*8*	*17*	D7
DØ	*9*	*16*	D6
D1	*10*	*15*	D5
D2	*11*	*14*	D4
Gnd	*12*	*13*	D3

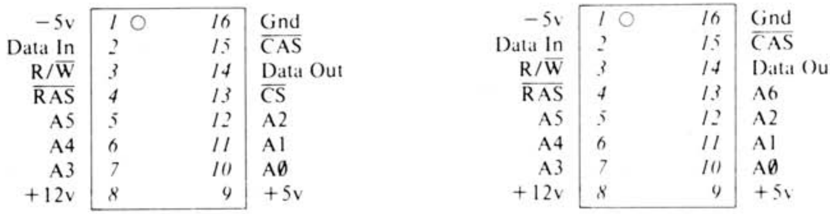
Kuva 13. ROM-piirin 9316B piikkijärjestys

RAM-MUISTI

Apple käyttää 4K ja 16K RAMeja pääasiallisena RAM-muistivarastonaan. RAM-muistia käyttävät sekä mikroprosessori että videonäyttöpiirit. Mikroprosessori ja videonäyttö vuorottelevat RAMin käytössä: mikroprosessori lukee tai kirjoittaa RAMiin vain Φ Ø:n aikana ja videonäyttö puhdistaa näytön RAMin Φ1:n aikana.

Kolme 74LS153:a osoitteissa E11, E12 ja E13, 74LS283 osoitteessa E14 ja puolet 74LS257:stä osoitteessa C12 muodostavat RAM-muistin osoitteiden multiplexerin. Ne ottavat mikroprosessorin ja videogeneraattorin muodostamat osoitteet ja multiplexaavat ne kuudelle RAMin osoitelinjalle. Muut RAM-osoitinsignaalit, $\overline{RAS}$ ja $\overline{CAS}$ sekä signaali, joka on osoitelinja 6 16K-RAMeille ja $\overline{CS}$ 4K-RAMeille, generoidaan RAM-valintapiirillä. Tämä piiri koostuu kahdesta 74LS139:stä osoitteissa E2 ja F2, 74LS153; puolikkaasta osoitteesta C1, puolestatoista 74LS257:stä osoitteissa C12 ja J1 sekä kolmesta muistin valintaliittimestä osoitteissa D1, E1 ja F1. Tämä piiri johtaa signaalit jokaiseen RAM-riviin riippuen siitä, minkä tyypin RAMia (4K vai 16K) ao. rivissä on.

RAMin data-sisäänmenot otetaan suoraan systeemin data-väylästä. Data-ulostulot lukitaan kahdella 74LS174:llä, jotka ovat osoitteissa B5 ja B8 ja multiplexataan näppäimistöltä tulevalla seitsemällä databitillä. Nämä lukitut RAM-ulostulot syötetään suoraan videogeneraattorin merkki-, väri- ja pistegeneraattoreihin, sekä myös takaisin systeemin data-väylään kahden osoitteissa B6 ja B7 olevan 74LS257:n avulla.



4096 4K RAM piikkijärjestys

4116 16K RAM piikkijärjestys

Kuva 14. RAM-piirien kytkentä

VIDEOGENERAATTORI

Näyttö on 192-juovainen, ja se on ryhmitelty 24:ään kahdeksan juovan ryhmään. Jokainen juova esittää neljänkymmenen tavun sisällön osittain tai kokonaan.

Videogenerointipiirit saavat tahdistus- ja ajastussignaalinsa 74LS161-laskureiden muodostamalta ketjulta joka on osoitteissa D11—D14. Nämä laskurit tuottavat viisitoista synkronointisignaalia:

HØ H1 H2 H3 H4 H5
VØ V1 V2 V3 V4
VA VB VC

Signaalien H-perhe ilmaisee tavun sarakkeen näytöllä ja ulottuu 000000 binaariseen 100111 (desimaalisena 39). Signaalit V0—V4 ilmoittavat rivin aseman näytössä. Se voi vaihdella 00000:sta binaariseen 10111:een (desimaalisena 23) VA, VB ja VC-signaalit ovat juovan asema rivissä, vaihdellen 000:sta binaariseen 111:een (desimaalisena 7).

Nämä signaalit lähetetään RAMin osoite-multiplexerille, joka kääntää ne yhden muistipaikan osoitteeksi, joka riippuu näytön pehmeiden kytkinten asennoista (ks. edellä). RAM-multiplexri lähettää sitten osoitteet vektoriin RAM-muistiin Φ1:n aikana. RAM-vektorin lähettämää datatietoa pidättävät lukot kääntävät sen videogeneraattoripiiriin. Muistipaikassa oleva 74LS283 uudelleenjärjestää osoitteet niin, että muistikartta menee sekaisin.

Jos näytöllä olevan alueen on tarkoitus olla merkki, videogeneraattori siirtää datan alimmat kuusi bittiä 2513-tyyppiseen generaattoriin, joka sijaitsee osoitteessa A5. Jokaisen merkin seitsemän riviä juovitetaan VA-, VB- ja VC-signaaleilla ja merkkigeneraattorin tulostus jaksotetaan pistevirraksi osoitteessa A3 olevan 74166:n avulla. Tämä bittivirta ohjataan eksklusiivisen OR-portin läpi, missä se käännetään, jos korkein bitti on sammutettu ja joko kuudes bitti on sammutettu tai osoitteessa B3 sijaitseva ajastin on ylhäällä. Tämä tuottaa käänteisen ja vilkkuvan tekstitilan. Tekstibittivirta lähetetään sitten valitsijalle/multiplexerille kuten edellä.

Jos Applen näyttö on grafiikkatilassa, RAMin data-tieto lähetetään kahdelle osoitteissa B4 ja B9 sijaitsevalle 74LS194 siirtorekisterille. Siellä jokainen tavunpuolikas muutetaan jaksotetuksi datavirraksi. Nämä kaksi datavirtaa lähetetään videovalitsimelle/multiplexerille.

Osoitteessa A8 oleva multiplexeri valitsee värin ja tarkkuusgrafiikan välillä. Osoitteessa A11 oleva 74LS74 viivyttää pistevirtaa puolen kellojakson ajan jos korkein bitti on sammutettu. Tämä tuottaa tarkkuusgrafiikan värit.

Videovalitsin/multiplexeri sekoittaa nämä edellä kuvatuista lähteistä tulevat kaksi datavirtaa näytön pehmeiden kytkimien asennon mukaisesti. 74LS194 osoitteessa A10 ja 74LS151 valitsevat yhden jaksotetuista bittijonoista tekstiä, värigrafiikkaa tai tarkkuusgrafiikkaa varten. Lopullinen jaksotettu ulostulo sekoitetaan yhdistetyllä synkronisaatiosignaalilla ja videosynkronisaatiopiirien tuottamalla väritahdistussignaalilla ja lähetetään video-ulostuloliittimille.

Videonäytön pehmeät kytkimet, jotka säätelevät näyttötiloja, dekoodataan Applen I/O-funktioiden avulla. Loogisia portteja osoitteissa B12, B13, B11, A12 ja A11 käytetään eri videotilojen kontrolloimiseen.

Väritahdistussignaali luodaan loogisilla porteilla osoitteissa B12, B13 ja C13 ja muokataan R5:llä, käämeillä L1 ja C2 sekä säätökondensaattorilla C3. Tällä säätökondensaattorilla voidaan näytön tuottamat värit säätää. Transistori Q6 ja sen kumppani resistori R27 sammuttavat väritahdistussignaalin kun Apple näyttää tekstiä.

VIDEOULOSTULOLIITTIMET

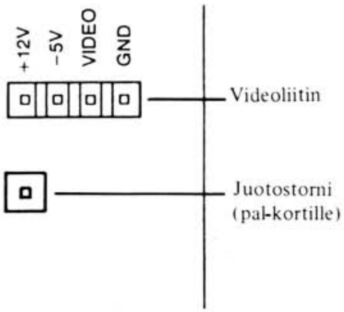
Edellä mainittujen piirien tuottama videosignaali on NTSC-kelpoinen, EIA-standardin mukainen, positiivinen yhdistetty videosignaali, joka voidaan syöttää mihin tahansa suljettuun videojärjestelmään tai videomonitoriin. Tämä signaali on saatavissa kolmessa paikassa Apple levyllä; RCA-liitin Applen levyn takaosassa lähellä oikeaa laitaa on RCA-standardin mukainen liitin. Tämän jakin pistukka on yhdistetty Applen tavalliseen maadoitukseen ja sen kärki on liitetty videotulostussignaaliin 200 ohmin potentiometrin kautta. Tämä potentiometri voi sovittaa liittimen jännitteen 0:n ja yhden voltin välille.

Näytön apuliitäntä Applen levyn oikeassa laidassa lähellä takareunaa on Molex KK100-sarjan liitäntä neljällä neliömäisellä piikillä, jota ovat .25" korkeita ja .10" keskustasta. Tämä liitäntä tuottaa yhdistetyn videotulostuksen ja kaksi virtalähdejännitettä. Tätä liitäntää esittää kuvio 15.

Taulukko 28: Erillisen videoliittimen signaalikuvaus

Nasta	Nimi	Kuvaus
1	GROUND	Järjestelmän sähköinen maa; Ø volttia.
2	VIDEO	NTSC-kelpoinen positiivinen video-signaali. Mustataso on noin 0.75 volttia. Valkoisen taso noin 2.0V. Tahdistussignaalin taso 0V. Ulostulotaso ei ole säädettävissä. Tätä linjaa ei ole suojattu oikosulkua vastaan.
3	+12v	+12V jännite
4	−5v	− 5V jännite

Näytön apuliitin. Tämä yksinkertainen metallijohtoliitin näytön apuliitännän alapuolella tuottaa samaa videosignaalia kuin sekin. Tämä on tarkoitettu liitäntäpisteeksi Eurapple PAL/SECAM-muunninlevyille.



Kuva 15. Videoliittimien kytkentä

SISÄÄNRAKENNETTU I/O

Applen sisäänrakennetut I/O-funktiot on kartoitettu 128:aan muistipaikkaan alkaen $C000:sta. Applen levyllä 74LS138 osoitteesta F13, jota kutsutaan I/O-valitsimeksi, dekoodaa nämä 128 osoitetta ja mahdollistaa useita toimintoja.

74LS138:n aktivoi toinen samanlainen piiri osoitteessa H12 aina kun Applen osoiteväylä sisältää osoitteen väliltä $C000–$C0FF. I/O-valitsin jakaa tämän 256:n tavun alueen kahdeksaan 16-tavuiseen alueeseen jättäen alueen $C080–$C0FF huomiotta. Jokainen 74SL138:n ulostulo-linja tulee aktiiviseksi, kun siihen yhdistettyyn 16-tavuiseen alueeseen viitataan.

I/O-valitsimen Ø-linja suuntaa datan näppäimistöliitännältä RAM data-multiplexerille.

Linja 1 nollaa 74LS74:n osoitteessa R10. Se on näppäimistölippu.

Linja 2 kääntää puolet 74LS74:stä osoitteessa K13. Tämän kytkimen ulostulo on yhdistetty resistoriverkon kautta kasettiulostuloliittiimen kärkeen.

Linja 3 kääntää toisen puolikkaan 74LS74:stä osoitteessa K13. Tämän kytkimen ulostulo on yhdistetty kondensaattorin ja Darlington-vahvistinpiirin kautta Applen kaiutinliitäntään, joka on levyn oikeassa laidassa näppäimistön alla.

Linja 4 on kytketty suoraan pelien I/O-liittimen nastaan 5. Tämä nasta on utility $\overline{\text{C040}}$ $\overline{\text{strobe}}$.

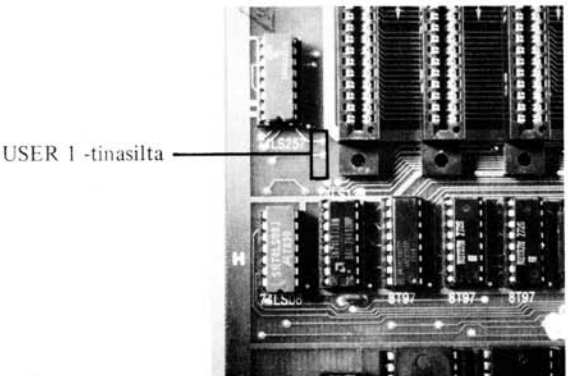
Linja 5 tekee 74LS259:n osoitteessa F14 aktiiviseksi. Tämä integroitu piiri sisältää videonäytön ja pelien I/O-liitännän ohajusulostulojen (annunciators) pehmeät kytkimet. Kytkimet valitaan osoitelinjoilla 1–3 ja jokaisen kytkimen asetusta valvoo osoituslinja 1.

Riviä 6 käytetään 74LS251 kahdeksanbittisen multiplexerin — joka sijaitsee osoitteessa R14 — toimivaksi saattamiseen. Tämä multiplexeri, toimivaksi tehtynä, yhdistää yhden kahdeksasta syöttörivistään kolmitilaisen järjestelmän dataväylän korkeimpaan bittiin (7:s). Kolme alinta riviä valvovat, minkä kahdeksasta sisäänmenosta multiplexeri valitsee. Neljä moninkertaistimen sisäämenoista tulee 553 ajastinneliköstä, osoitteesta H13. Ajastimen otot ovat pelien I/O-liitännän ohjausliittimiä. Multiplexerin kolme muuta sisäänmenoa tulevat yhden bitin sisäänmenoista (painonapit) pelien I/O-liitännästä. Viimeinen multiplexer-sisäänmeno 741 operaatiovahvistimesta osoitteesta K13. Tämän operaatiovahvistimen sisäämeno tulee kasettinauhurin sisäänmenoliittimestä.

I/O-valitsimen rivi 7 nollaa kaikki osoitteesa H13 olevan 553 ajastin nelikön neljä ajastinta. Tämän ajastimen neljä sisäänmenoa tulevat neljän 0.022 uF-kondensaattorin, neljän 100 ohmin resistorin ja pelien I/O-liittimeen kytkettyjen peliohjainten potentiometrien muodostamasta verkosta. Kokonaisresistanssi jokaisesta neljästä ajastuspiiristä määrää piirin ajastusominaisuudet.

USER 1 -LIITIN

Applen emolevyssä, vasemmalle väylä Ø:sta, on juotettu foliosiltapari, jota kutsutaan User 1 -liittimeksi. Tämä liitin on esitetty kuvassa 8. Jos näiden kahden foliosillan väliin kytkee johdon, jokaisen oheislaiteliitännän USER 1 -linja tulee aktiiviseksi. Jos mikä tahansa oheiskortti vetää tämän linjan alas, kaikki sisäinen I/O-dekoodaus katkeaa. I/OSELECT ja DEVICE SELECT -linjat ovat ylhäällä ja jäävät ylös kun USER 1 on alhaalla riippumatta osoiteväylän osoitteista.



Valokuva 8. USER 1 -tinasilta

PELIEN I/O-LIITTIMET

+5v	1 ○	16	NC
PBØ	2	15	ANØ
PB1	3	14	AN1
PB2	4	13	AN2
$\overline{\text{CØ4Ø}}$ $\overline{\text{STROBE}}$	5	12	AN3
GCØ	6	11	GC3
GC2	7	10	GC1
Gnd	8	9	NC

Kuvio 16.
Pelien I/O-liittimen kytkentä

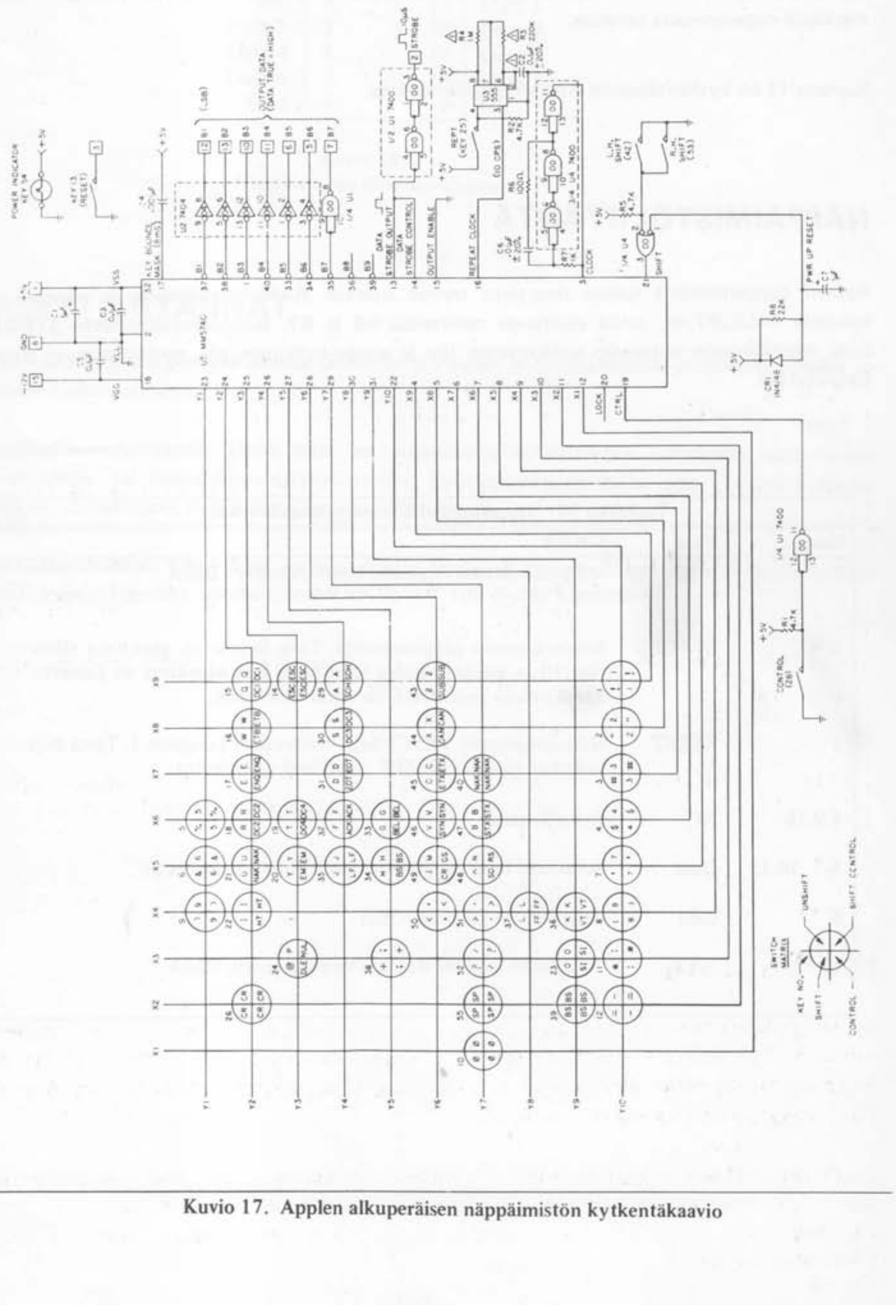
Taulukko 29: Pelien I/O-liittimen signaalikuvaus

Nasta:	Nimi:	Kuvaus:
1	+5v	+5 V jännite. Suurin ottovirta on oltava alla 100 mA.
2-4	PBØ-PB2	Yhden bitin (painonappi) sisäänmenot. Tavallisia 74LS-sarjan TTL-sisäänmenoja.
5	$\overline{\text{CØ4Ø}}$ $\overline{\text{STROBE}}$	Yleiskäyttöinen STROBE-signaali. On normaalisti looginen 1 (high), mutta muuttuu Ø:ksi kun suoritetaan luku tai kirjoitus-käsky mihin tahansa osoitteeseen välillä $CØ40–$CØ4F. Standardi 74LS TTL-ulostulo.
6,7,10,11	GCØ-GC3	Peliohjainten sisäänmenot. Kaikki nämä sisäänmenot on syytä kytkeä 150K ohmin potentiometrillä 5V:n.
8	Gnd	Järjestelmän sähköinen maa
12-15	ANØ-AN3	Ulostulot. Nämä standardit 74LS-sarjan TTL ulostulot on syytä puskuroida, mikäli ne kytketään muihin kuin TTL-sisäänmenoihin. inputs.
9,16	NC	Ei sisäistä kytkentää.

NÄPPÄIMISTÖ

Applen sisäänrakennettu näppäimistö on rakennettu MM5740 monoliittisen näppäimistödekooderi-ROMin ympärille. ROMille tulevat sisäänmenot-liitännöistä 4–12 ja 22–31 yhdistetään näppäimistön näppäinkytkinmatriisiin. ROMin ulostulot puskuroidaan 7404:ään ja yhdistetään Applen näppäimistöliitäntään (seuraavana).

Näppäimistö-dekooderi pyyhkäisee nopeasti näppäinmatriisin yli ja etsii, mitä näppäintä on painettu. Pyyhkäisytoimintaa kontrolloi levyosoitteessa U4 oleva vapaasti värähtelevä oskillaattori, joka koostuu kolmesta 7400:sta. Värähtelyn nopeutta valvotaan painokytkentälevyllä olevilla C6:lla, R6:lla ja R7:llä.



Kuvio 17. Applen alkuperäisen näppäimistön kytkentäkaavio

REPT-näppäin on yhdistetty 555-ajastinpiiriin, joka sijaitsee levyosoitteessa U3. Tämä piiri ja kondensaattori sekä sen ympärillä olevat kolme resistoria generoivat 10Hz:n REPeat-signaalin. Jos 220K ohmin resistori R3 korvataan pienempiarvoisella resistorilla, REPT-näppäin toistaa merkkejä nopeammassa tahdissa.

Kuvassa 17 on kytkentäkaavio Applen näppäimistöstä.

NÄPPÄIMISTÖLIITÄNTÄ

Applen näppäimistöltä tuleva data-tieto menee suoraan RAM-multiplexerin ja salpojen läpi kahdelle 74LS257:lle, jotka sijaitsevat osoitteissa B6 ja B7. Näppäimistöliitännän STROBE-linja, näppäimistön seitsemän sisäänmenon tila ja strobe-kytkimen tila, multiplexataan Applen data-väylään.

Taulukko 30: Näppäimistön liitännän signaalikuvaus

Nasta	Nimi	Kuvaus
1	+5v	+5 voltin jännite. Maksimi kuormitusvirta 120mA.
2	STROBE	Strobe-ulostulo näppäimistöltä. Tälle linjalle on annettava vähintäin 10 µs pitoinen pulssi joka kerta kun näppäintä on painettu. Tämä pulssin polariteetti on vapaavalintainen.
3	$\overline{\text{RESET}}$	Mikroprosessorin RESET-linja. Normaalisti Looginen 1. Tämä linja vedetään alas kun RESET näppäintä on painettu.
4,9,16	NC	Ei kytkentää.
5-7, 10-13	Data	Seitsemän bitin näppäimistön ASCII-data sisäänmeno.
8	Gnd	Järjestelmän sähköinen maa.
15	−12v	−12 voltin jännite. Maksimi kuormitusvirta 50mA.

+5v	*1* ○	*16*	NC
STROBE	*2*	*15*	−12v
RESET	*3*	*14*	NC
NC	*4*	*13*	Data 1
Data 5	*5*	*12*	Data Ø
Data 4	*6*	*11*	Data 3
Data 6	*7*	*10*	Data 2
Gnd	*8*	*9*	NC

Kuvio 18.
Näppäimistön liittimen kytkentä

KASETTILIITÄNNÄT

Kaksi naaraspuolista pienoiskokoista jakkiliitintä Applen levyn takaosassa yhdistävät Applen tavalliseen kasettinauhuriin.

Kasettisisäänmenoliitäntä: Tämä liitin on suunniteltu kytkettäväksi useimpien nauhureiden korvakuuloke- tai lisäkaiutin-ulostuloliittimiin. Syöttöjännitteen tulisi olla 1 voltti huipusta huippuun (nimellinen). Syöttöimpedanssi on 12 KOhmia.

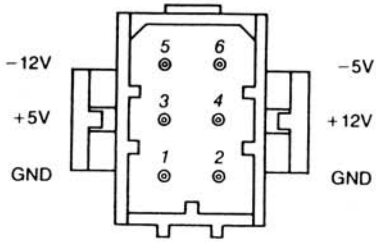
Kasettiulostuloliitin: Tämä liitin on suunniteltu kytkettäväksi tavallisen kasettinauhurin mikrofonisisäämenoliitäntään. Ulostulojännite on 25 mV 100 ohmin kuormaan.

VIRTALIITÄNTÄ

Tähän liitäntään yhdistetään Applen virtalähteen kaapeli. Se on AMP 9-35028-1 kuusinapainen urospuolinen liitin.

Taulukko 31: Virtalähdeliittimen kytkentä

Nasta	Nimi	Kuvaus
1,2	Maa	Applen jännitteiden yhteinen, sähköinen maa.
3	+5v	+ 5.0 voltin jännite virtalähteeltä. Apple II varustettuna 48K:n RAM-muistilla, ilman oheislaitteita, kuormittaa tätä jännitettä 1.5A.
4	+12v	+ 12 voltin jännite virtalähteeltä. Apple II varustettuna 48K:n muistilla, ilman oheislaitteita kuormittaa tätä jännitettä ~400 mA.
5	−12v	− 12 voltin jännite virtalähteeltä. Apple II varustettuna 48K:n muistilla, ilman oheislaitteita kuormittaa tätä jännitettä ~12,5mA.
6	−5v	− 5 voltin jännite virtalähteeltä. Apple II varustettuna 48K:n muistilla, ilman oheislaitteita, kuormittaa tätä jännitettä 0,0mA.



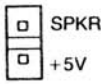
Kuva 19. Virtalähdeliitin

KAIUTIN

Applen sisäistä kaiutinta käytetään 74LS74-kytkimen puolikkaalla Darlington-vahvistimen läpi. Kaiutinliitäntä on Molex KK100-sarjan liitäntä, jossa on kaksi nastaa, .25" korkeita, .10" läpimitaltaan.

Taulukko 32: Kaiuttimen kytkentä ja ulostulosignaali

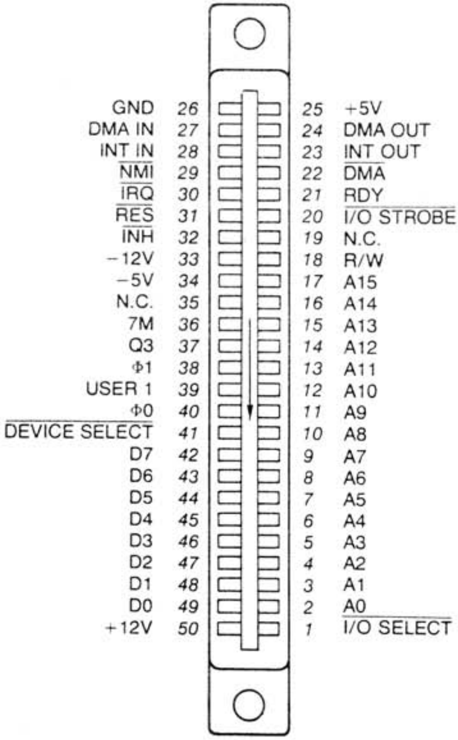
Nasta:	Nimi:	Kuvaus:
1	SPKR	Kaiutinsignaali. Ulostulo antaa noin 0.5W 8 ohmin kuormaan.
2	+5v	+5V jännite virtalähteeltä.



Kuva 20. Kaiutinliitin

OHEISLAITELIITÄNNÄT

Kahdeksan oheisliitäntää Applen levyn takareunassa ovat Winchester 2HW2500-111 50-napaisia kortinlaitaliittimiä .10" halkaisijaltaan olevin nastoin. Näiden liitäntöjen kytkennät esitetään kuvassa 21 ja signaalikuvaukset annetaan seuraavilla sivuilla.



Kuva 21. Oheislaiteliittimen kytkentä

Taulukko 33: Oheislaiteliitäntöjen signaalikuvaus

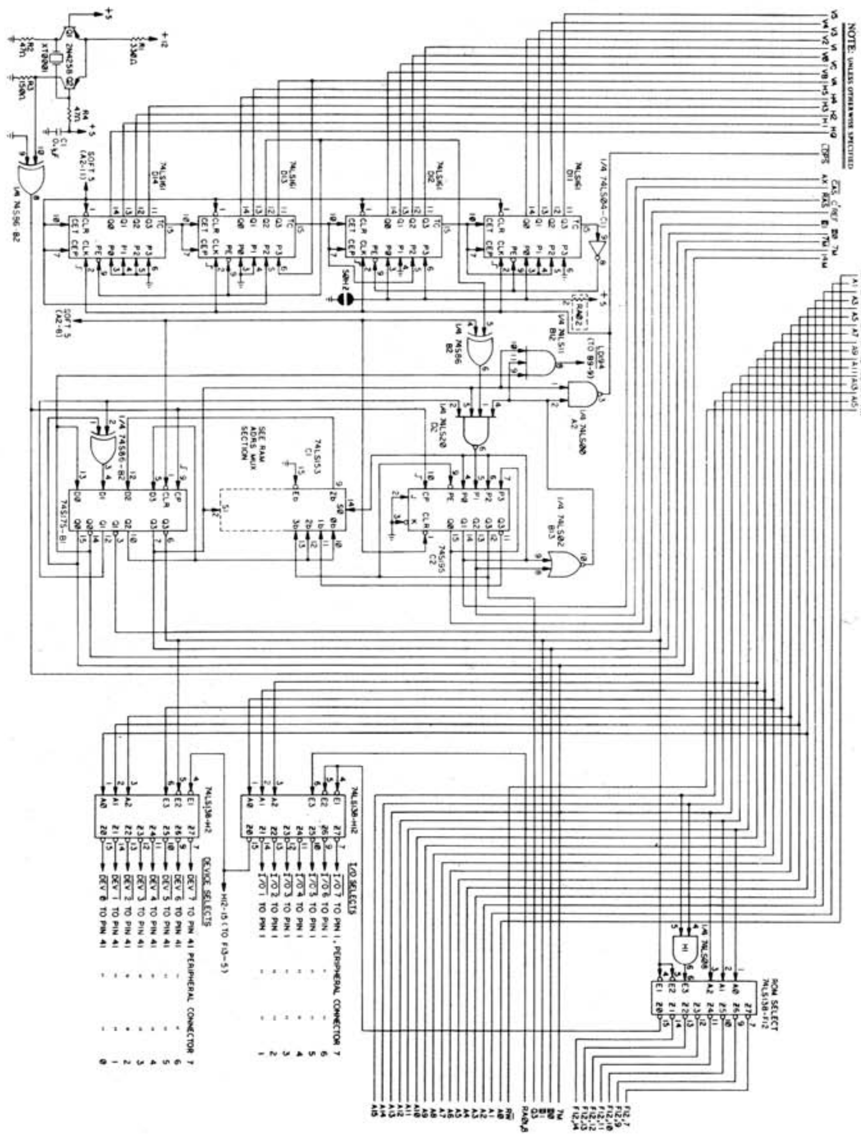
Nasta	Nimi	Kuvaus
1	$\overline{\text{I/O SELECT}}$	Tämä normaalisti 1-tilassa oleva linja laskee nollaksi, kun mikroprosessori kutsuu sivua $Cn, jossa n = jonkin väylän numero. Signaali tulee aktiiviseksi kellojakson Φ Ø aikana ja pystyy ohjaamaan kymmentä LS TTL kuormaa*. Tämä signaali ei ole käytettävissä väylässä n:o Ø.
2-17	AØ-A15	Puskuroitu osoiteväylä. Osoite näissä linjoissa tulee käyttökelpoiseksi kellojakson Φ1 aikana ja pysyy yllä kellojakson Φ Ø ajan. Nämä linjat pystyvät kukin ohjaamaan viittä LSTTL-kuormaa*.
18	R/$\overline{\text{W}}$	Puskuroitu luku/$\overline{\text{kirjoitus}}$-signaali. Tämä linja tulee käyttökelpoiseksi samalla kun osoiteväyläkin; nousee 1:ksi lukujakson aikana ja laskee Ø:ksi kirjoitusjaksolla. Tämä linja kykenee ohjaamaan kahta LSTTL-kuormaa*.
19	SYNC	Vain oheislaiteliittimessä 7. Tämä nasta on kytketty video ajastingeneraattorin SYNC-signaaliin.
20	$\overline{\text{I/O STROBE}}$	Tämä linja laskee kellojakson Φ Ø aikana, jolloin osoiteväylässä on jokin osoite välillä $C800 $CFFF. Tämä linja kykenee ohjaamaan neljää LSTTL-kuormaa*.
21	RDY	6502:n RDY-sisäänmeno. Tämän linjan vetäminen alas kellojakson Φ1 aikana pysäyttää prosessorin toiminnan, jättäen osoiteväylään juuri käsiteltävänä olleen osoitteen.
22	$\overline{\text{DMA}}$	Tämän linjan alasvetäminen passivoi 6502:n osoiteväylän ja pysäyttää prosessorin toiminnan. Tämä linja on kytketty 3K Ω:n vastuksella virtalähteen +5V.
23	INT OUT	Daisy-chained -keskeytysulostulo vähemmän tärkeille oheislaitteille. Tämä nasta on yleensä kytketty nastaan 28 (INT IN).
24	DMA OUT	Daisy chained -DMA-ulostulo vähemmän tärkeille oheislaitteille. Tämä nasta on yleensä kytketty nastaan 22 (DMA IN).
25	+5v	+5 voltin jännite virtalähteeltä. Max 500mA:n virta on saatavissa tästä liittimestä yhteisesti. Kaikille liitäntäkorteille.
26	GND	Järjestelmän sähköinen maa.

* Kuormitukset kullekin oheislaitekortille.

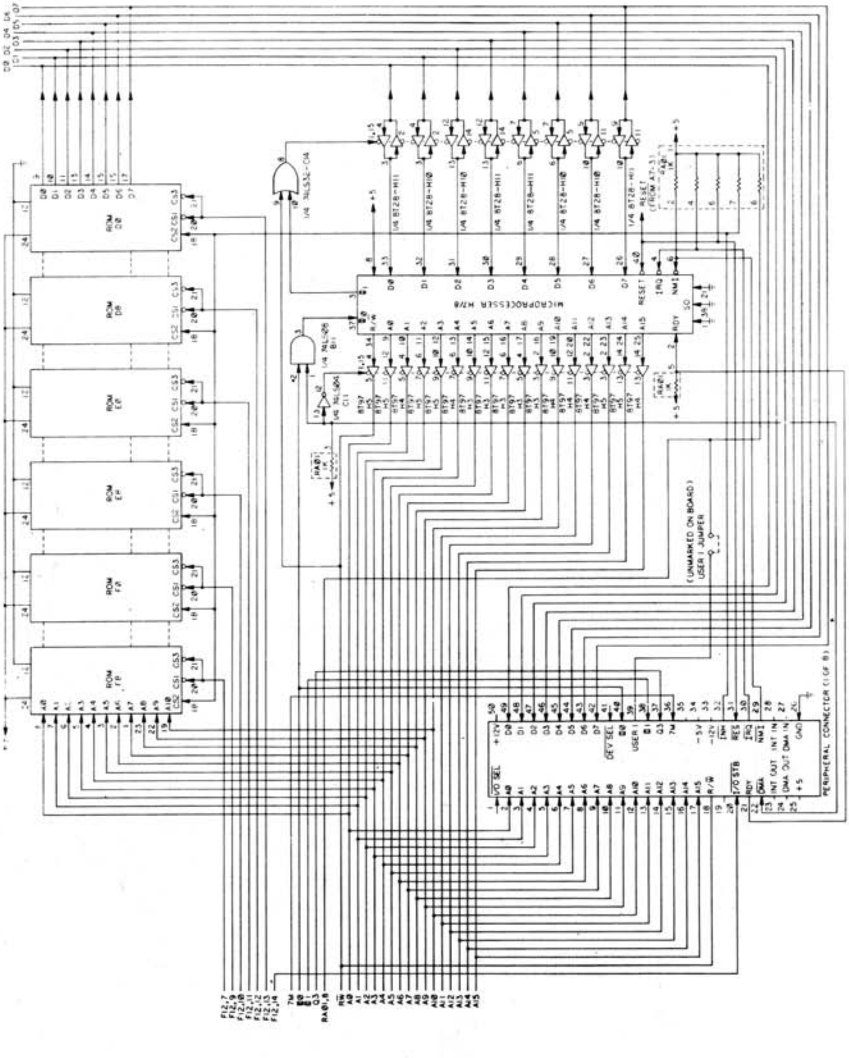
** Katso sivu 99.

Taulukko 33 (jatkoa): Oheislaiteliitäntöjen signaalikuvaus		
Nasta	Nimi	Kuvaus
27	DMA IN	Daisy-chained DMA-sisäänmeno ylempiarvoisemmilta laitteilta kytketty normaalisti nastaan 24 (DMA OUT).
26	INT IN	Daisy-chained -keskeytyssisäänmeno ylempiarvoisimmilta laitteilta. Kytketty normaalisti nastaan 23 (INT OUT).
29	$\overline{\text{NMI}}$	NMI-keskeytyspyyntö. Kun tämä linja vedetään alas, aloittaa Apple keskeytysjakson ja hyppää muistiosoitteen $3FB osoittamaan keskeytyspalvelualiohjelmaan.
30	$\overline{\text{IRQ}}$	IRQ-keskeytyspyyntö. Kun tämä linja vedetään alas, aloittaa Apple keskeytysjakson sillä ehdolla, että 6502:n I-lippu (Interrupt disable) ei ole asetettu. Mikäli keskeytys suoritetaan, hypätään muistiosoitteiden $3FE ja $3FF osoittamaan keskeytyspalvelualiohjelmaan.
31	$\overline{\text{RES}}$	Kun tämä linja vedetään alas, aloittaa Apple nollausjakson (RESET-cycle) (ks. sivu 36).
32	$\overline{\text{INH}}$	Kun tämä linja vedetään alas, kytkeytyvät kaikki Applen emolevyn ROM-piirit pois järjestelmän muistiavaruudesta. Linja on kytketty 3KΩ vastuksella +5V:iin.
33	−12v	− 12 voltin jännite. Maksimivirta kaikille oheislaitteille on yhteensä 200mA.
34	−5v	− 5 voltin jännite. Maksimivirta kaikille oheislaitteille on yhteensä 200mA.
35	COLOR REF	Vain väyläliittimessä 7. Tämä linja on kytketty videogeneraattorin 3.5MHz COLOR REF -signaaliin.
36	7M	7MHz:n kellotaajuus. Tämä linja kykenee ohjaamaan kahta LSTTL-kuormaa.*
37	Q3	2MHz:n asymmetrinen kellotaajuus. Tämä linja kykenee ohjaamaan kahta LSTTL-kuormaa.*
38	Φ1	Mikroprosessorin kellotaajuuden vaihe yksi. Tämä linja kykenee ohjaamaan kahta LSTTL-kuormaa.*
39	USER 1	Kun tämä linja vedetään alas, estyy kaikki sisäinen I/O-osoitekoodaus. **

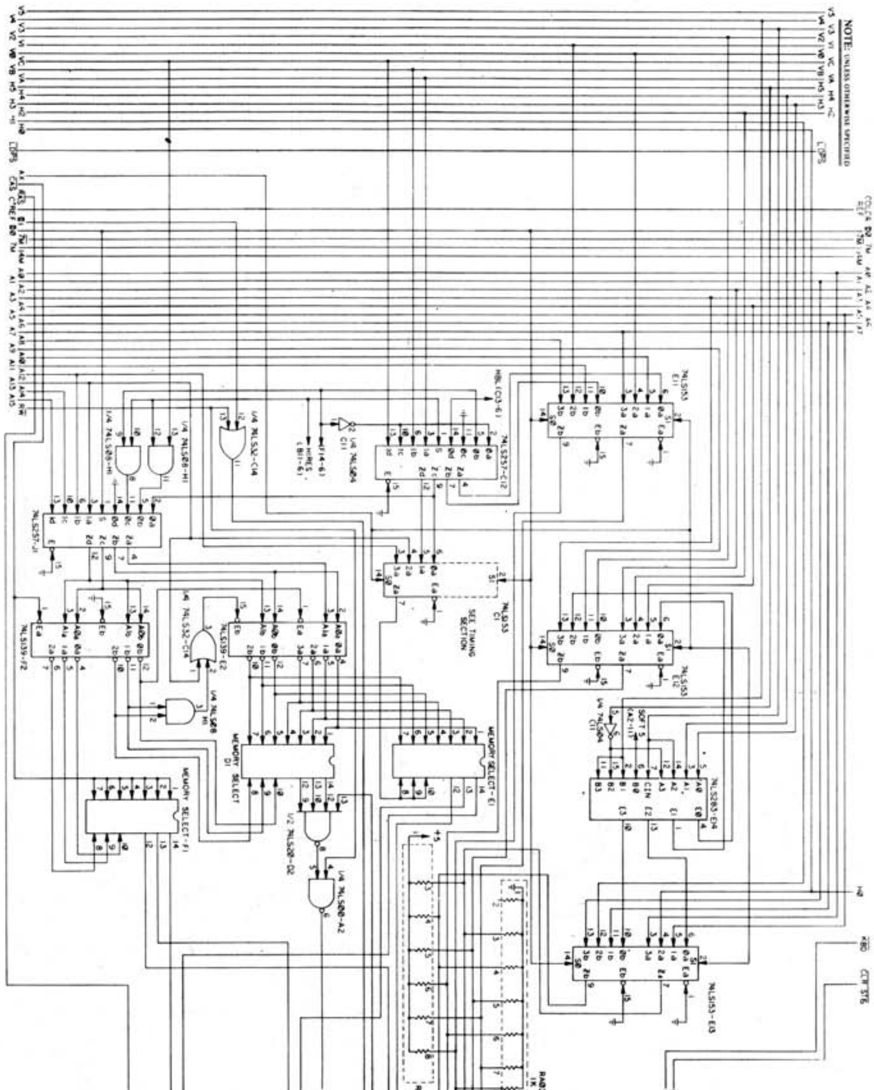
Taulukko 33 (jatkoa): Oheislaiteliitäntöjen signaalikuvaus		
Nasta	Nimi	Kuvaus
40	ΦØ	Mikroprosessorin kellotaajuuden vaihe nolla. Tämä linja kykenee ohjaamaan kahta LSTTL-kuormaa.*
41	$\overline{\text{DEVICE SELECT}}$	Tämä linja tulee aktiiviseksi (nolla), kun prosessorin osoiteväylässä on osoite välillä $CØnØ–$CØnF, jossa n on ko. väylän numero plus $8. Tämä linja kykenee ohjaamaan 10 LSTTL-kuormaa.*
42-49	DØ-D7	Puskuroitu kaksisuuntainen dataväylä. Data tässä väylässä on tosi 3ØØnS ajan kirjoitusjakson Φ Ø aikana ja pysyy stabiilina aina 100nS ennen jakson Φ Ø loppua, lukujaksolla. Kukin linja kykenee ohjaamaan yhtä LSTTL-kuormaa.
50	+12v	+12 voltin jännite. Tämä linja kykenee syöttämään yhteensä 250mA kaikille oheislaitekorteille.



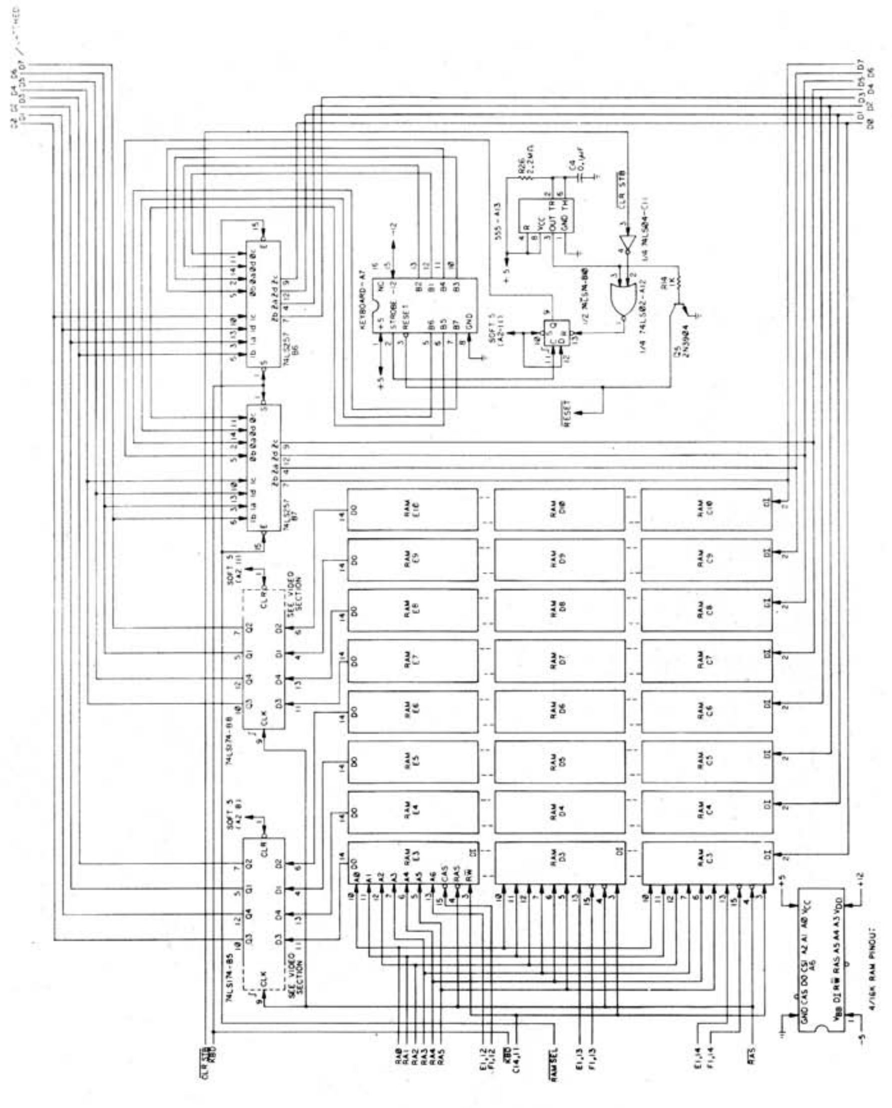
Kuva 22-1. Apple II:n kytkentäkaavio



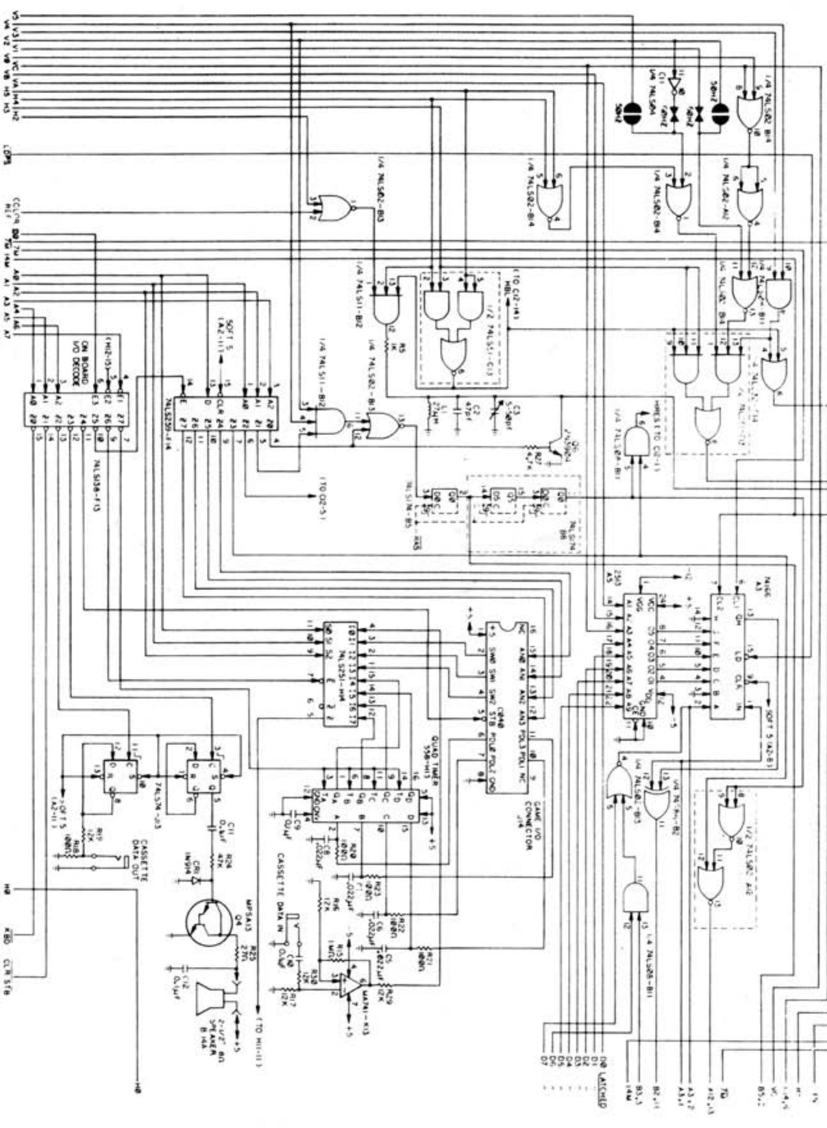
Kuva 22-2. Apple II:n kytkentäkaavio



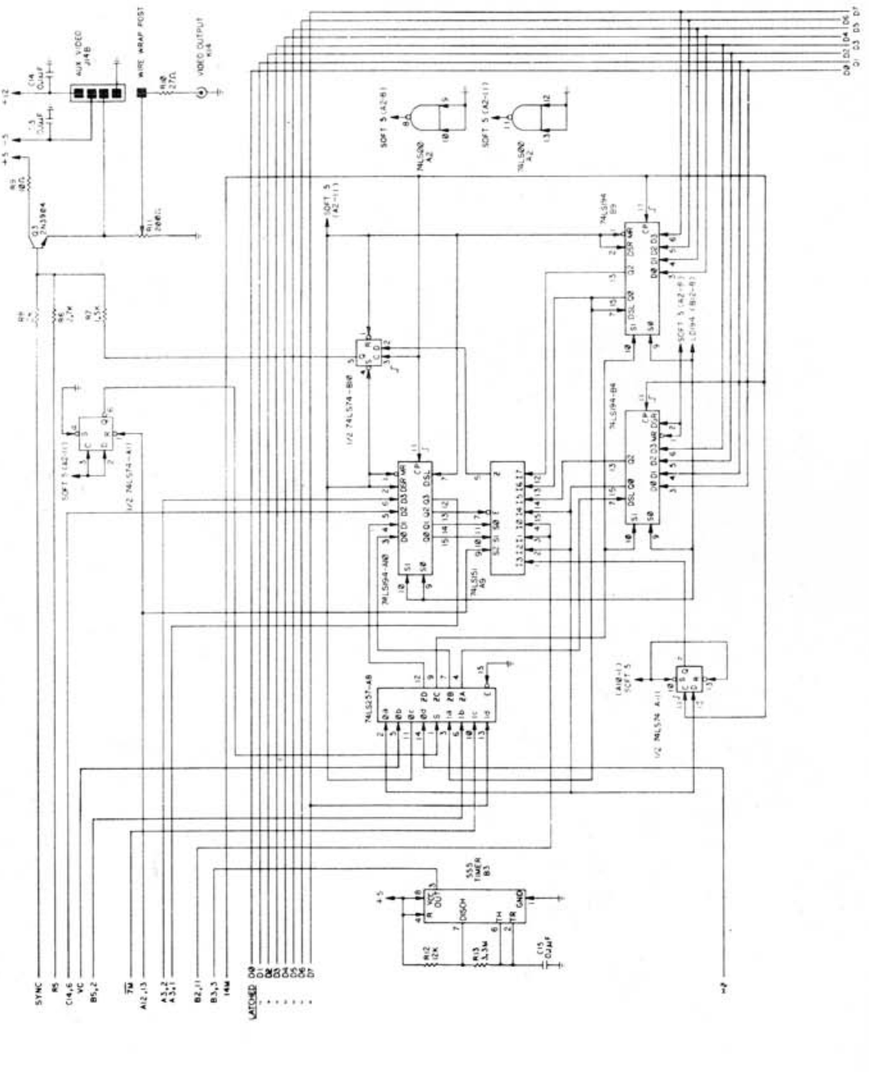
Kuva 22-3. Apple II:n kytkentäkaavio



Kuva 22-4. Apple II:n kytkentäkaavio



Kuva 22-5. Apple II:n kytkentäkaavio



Kuva 22-6. Apple II:n kytkentäkaavio

LIITE A
6502-PROSESSORIN KÄSKYKANTA

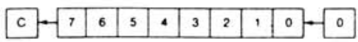
6502 PROSESSORIN KÄSKYKANTA

ADC	Add Memory to Accumulator with Carry
AND	"AND" Memory with Accumulator
ASL	Shift Left One Bit (Memory or Accumulator)
BCC	Branch on Carry Clear
BCS	Branch on Carry Set
BEQ	Branch on Result Zero
BIT	Test Bits in Memory with Accumulator
BMI	Branch on Result Minus
BNE	Branch on Result not Zero
BPL	Branch on Result Plus
BRK	Force Break
BVC	Branch on Overflow Clear
BVS	Branch on Overflow Set
CLC	Clear Carry Flag
CLD	Clear Decimal Mode
CLI	Clear Interrupt Disable Bit
CLV	Clear Overflow Flag
CMP	Compare Memory and Accumulator
CPX	Compare Memory and Index X
CPY	Compare Memory and Index Y
DEC	Decrement Memory by One
DEX	Decrement Index X by One
DEY	Decrement Index Y by One
EOR	"Exclusive-Or" Memory with Accumulator
INC	Increment Memory by One
INX	Increment Index X by One
INY	increment Index Y by One
JMP	Jump to New Location
JSR	Jump to New Location Saving Return Address
LDA	Load Accumulator with Memory
LDX	Load Index X with Memory
LDY	Load Index Y with Memory
LSR	Shift Right one Bit (Memory or Accumulator)
NOP	No Operation
ORA	"OR" Memory with Accumulator
PHA	Push Accumulator on Stack
PHP	Push Processor Status on Stack
PLA	Pull Accumulator from Stack
PLP	Pull Processor Status from Stack
ROL	Rotate One Bit Left (Memory or Accumulator)
ROR	Rotate One Bit Right (Memory or Accumulator)
RTI	Return from Interrupt
RTS	Return from Subroutine
SBC	Subtract Memory from Accumulator with Borrow
SEC	Set Carry Flag
SED	Set Decimal Mode
SEI	Set Interrupt Disable Status
STA	Store Accumulator in Memory
STX	Store Index X in Memory
STY	Store Index Y in Memory
TAX	Transfer Accumulator to Index X
TAY	Transfer Accumulator to Index Y
TSX	Transfer Stack Pointer to Index X
TXA	Transfer Index X to Accumulator
TXS	Transfer Index X to Stack Pointer
TYA	Transfer Index Y to Accumulator

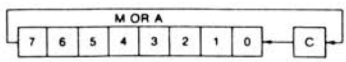
SEURAAVASSA YHTEENVEDOSSA KÄYTETYT LYHENTEET JA TOIMINTOJEN SELOSTUS

A	Akku
X, Y	Indeksirekisterit
M	Muisti
$\bar{C}$	Lainaus
P	Prosessorin status rekisteri
S	Pino-osoitin
✓	Muutos
—	Ei muutosta
+	Lisäys
∧	Looginen AND
-	Vähennyslasku
⊻	Looginen Eksklusiivinen OR
↑	Siirtyy pinosta (stack)
↓	Siirtyy pinoon (stack)
→	Siirtyy nuolen suuntaan
←	Siirtyy nuolen suuntaan
V	Looginen OR
PC	Ohjelmalaskuri
PCH	Ohjelmalaskurin ylempi tavu
PCL	Ohjelmalaskurin alempi tavu
OPER	Operandi
#	Suora osoitusmuoto

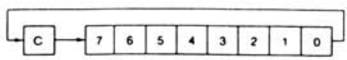
KUVA 1 Aritmeettinen siirto vasemmalle (ASL), yhden bitin operaatio



KUVA 2 Pyöritä yksi bitti vasemmalle (Muisti tai Akku)



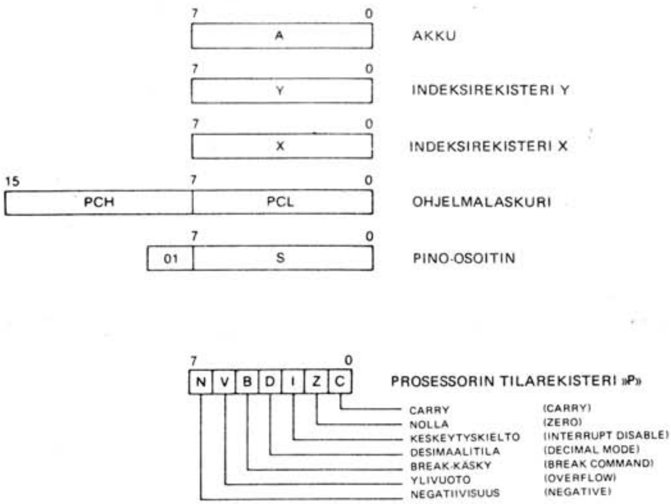
KUVA 3



HUOM 1: BIT – testaa bitit

Bitti 6 ja 7 siirretään tilarekisteriin. Jos tuloksena operaatiosta A M saadaan nolla, niin Z = 1, muutoin Z = Ø

OHJELMOINTIMALLI



KÄSKYKANTA

Nimi Kuvaus	Toiminto	Osoitus-muoto	Assembly-kielinen muoto	HEX OP-koodi	Tilan-tarve bytea	»P» status rekisteri N Z C I D V
ADC Add memory to accumulator with carry	A+M+C → A,C	Immediate	ADC #Oper	69	2	√√√- - √
		Zero Page	ADC Oper	65	2	
		Zero Page,X	ADC Oper,X	75	2	
		Absolute	ADC Oper	6D	3	
		Absolute,X	ADC Oper,X	7D	3	
		Absolute,Y	ADC Oper,Y	79	3	
		(Indirect,X)	ADC (Oper,X)	61	2	
		(Indirect),Y	ADC (Oper),Y	71	2	
AND "AND" memory with accumulator	A ∧ M → A	Immediate	AND #Oper	29	2	√√- - - -
		Zero Page	AND Oper	25	2	
		Zero Page,X	AND Oper,X	35	2	
		Absolute	AND Oper	2D	3	
		Absolute,X	AND Oper,X	3D	3	
		Absoldte,Y	AND Oper,Y	39	3	
		(Indirect,X)	AND (Oper,X)	21	2	
		(Indirect),Y	AND (Oper),Y	31	2	
ASL Shift left one bit (Memory or Accumulator)	(See Figure 1)	Accumulator	ASL A	0A	1	√√√- - -
		Zero Page	ASL Oper	06	2	
		Zero Page,X	ASL Oper,X	16	2	
		Absolute	ASL Oper	0E	3	
		Absolute,X	ASL Oper,X	1E	3	
BCC Branch on carry clear	Branch on C=0	Relative	BCC Oper	90	2	- - - - - -
BCS Branch on carry set	Branch on C=1	Relative	BCS Oper	B0	2	- - - - - -
BEQ Branch on result zero	Branch on Z=1	Relative	BEQ Oper	F0	2	- - - - - -
BIT Test bits in memory with accumulator	A ∧ M, M_7 → N, M_6 → V	Zero Page	BIT* Oper	24	2	M_7√- - - M_6
		Absolute	BIT* Oper	2C	3	
BMI Branch on result minus	Branch on N=1	Relative	BMI Oper	30	2	- - - - - -
BNE Branch on result not zero	Branch on Z=0	Relative	BNE Oper	D0	2	- - - - - -
BPL Branch on result plus	Branch on N=0	Relative	BPL oper	10	2	- - - - - -
BRK Force Break	Forced Interrupt PC+2 ↓ P ↓	Implied	BRK*	00	1	- - - 1 - -
BVC Branch on overflow clear	Branch on V=0	Relative	BVC Oper	50	2	- - - - - -

Note 1: [illegible] 6 and 7 are transferred to the status register. If the result of A ∧ M is [illegible] then [illegible] 1, otherwise Z = 0

Note 2: A BRK command cannot be masked by setting I

Nimi Kuvaus	Toiminto	Osoitus- muoto	Assembly- kielinen muoto	HEX OP- koodi	Tilan- tarve bytes.	»P» status rekisteri N Z C I D V
BVS Branch on overflow set	Branch on V=1	Relative	BVS Oper	70	2	------
CLC Clear carry flag	0 → C	Implied	CLC	18	1	---0--
CLD Clear decimal mode	0 → D	Implied	CLD	D8	1	-0----
CLI	0 → I	Implied	CLI	58	1	---0--
CLV Clear overflow flag	0 → V	Implied	CLV	B8	1	0-----
CMP Compare memory and accumulator	A — M	Immediate Zero Page Zero Page, X Absolute Absolute,X Absolute,Y (Indirect,X) (Indirect),Y	CMP #Oper CMP Oper CMP Oper,X CMP Oper CMP Oper,X CMP Oper,Y CMP (Oper,X) CMP (Oper),Y	C9 C5 D5 CD DD D9 C1 D1	2 2 2 3 3 3 2 2	√√√---
CPX Compare memory and index X	X — M	Immediate Zero Page Absolute	CPX #Oper CPX Oper CPX Oper	E0 E4 EC	2 2 3	√√√---
CPY Compare memory and index Y	Y — M	Immediate Zero Page Absolute	CPY #Oper CPY Oper CPY Oper	C0 C4 CC	2 2 3	√√√---
DEC Decrement memory by one	M — 1 → M	Zero Page Zero Page,X Absolute Absolute,X	DEC Oper DEC Oper,X DEC Oper DEC Oper,X	C6 D6 CE DE	2 2 3 3	√√----
DEX Decrement index X by one	X — 1 → X	Implied	DEX	CA	1	√√----
DEY Decrement index Y by one	Y — 1 → Y	Implied	DEY	88	1	√√----

Nimi Kuvaus	Toiminto	Osoitus-muoto	Assembly-kielinen muoto	HEX OP-koodi	Tilan tarve byteä	aPa status rekisteri N Z C I D V
EOR						
"Exclusive-Or" memory with accumulator	A ∀ M → A	Immediate	EOR #Oper	49	2	√√----
		Zero Page	EOR Oper	45	2	
		Zero Page,X	EOR Oper,X	55	2	
		Absolute	EOR Oper	4D	3	
		Absolute,X	EOR Oper,X	5D	3	
		Absolute,Y	EOR Oper,Y	59	3	
		(Indirect,X)	EOR (Oper,X)	41	2	
		(Indirect),Y	EOR (Oper),Y	51	2	
INC						
Increment memory by one	M + 1 → M	Zero Page	INC Oper	E6	2	√√----
		Zero Page,X	INC Oper,X	F6	2	
		Absolute	INC Oper	EE	3	
		Absolute,X	INC Oper,X	FE	3	
INX						
Increment index X by one	X + 1 → X	Implied	INX	E8	1	√√----
INY						
Increment index Y by one	Y + 1 → Y	Implied	INY	C8	1	√√----
JMP						
Jump to new location	(PC+1) → PCL	Absolute	JMP Oper	4C	3	------
	(PC+2) → PCH	Indirect	JMP (Oper)	6C	3	
JSR						
Jump to new location saving return address	PC+2 ↓, (PC+1) → PCL (PC+2) → PCH	Absolute	JSR Oper	20	3	------
LDA						
Load accumulator with memory	M → A	Immediate	LDA #Oper	A9	2	√√----
		Zero Page	LDA Oper	A5	2	
		Zero Page,X	LDA Oper,X	B5	2	
		Absolute	LDA Oper	AD	3	
		Absolute,X	LDA Oper,X	BD	3	
		Absolute,Y	LDA Oper,Y	B9	3	
		(Indirect,X)	LDA (Oper,X)	A1	2	
		(Indirect),Y	LDA (Oper),Y	B1	2	
LDX						
Load index X with memory	M → X	Immediate	LDX #Oper	A2	2	√√----
		Zero Page	LDX Oper	A6	2	
		Zero Page,Y	LDX Oper,Y	B6	2	
		Absolute	LDX Oper	AE	3	
		Absolute,Y	LDX Oper,Y	BE	3	
LDY						
Load index Y with memory	M → Y	Immediate	LDY #Oper	A0	2	√√----
		Zero Page	LDY Oper	A4	2	
		Zero Page,X	LDY Oper,X	B4	2	
		Absolute	LDY Oper	AC	3	
		Absolute,X	LDY Oper,X	BC	3	

Nimi Kuvaus	Toiminto	Osoitus- muoto	Assembly- kielinen muoto	HEX OP- koodi	Tilan- tarve byteä	»P» status rekisteri » N Z C I D V
LSR Shift right one bit (memory or accumulator)	(See Figure 1)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	LSR A LSR Oper LSR Oper,X LSR Oper LSR Oper,X	4A 46 56 4E 5E	1 2 2 3 3	0√√---
NOP No operation	No Operation	Implied	NOP	EA	1	------
ORA "OR" memory with accumulator	A V M → A	Immediate Zero Page Zero Page,X Absolute Absolute,X Absolute,Y (Indirect,X) (Indirect),Y	ORA #Oper ORA Oper ORA Oper,X ORA Oper ORA Oper,X ORA Oper,Y ORA (Oper,X) ORA (Oper),Y	09 05 15 0D 1D 19 01 11	2 2 2 3 3 3 2 2	√√----
PHA Push accumulator on stack	A ↓	Implied	PHA	48	1	------
PHP Push processor status on stack	P ↓	Implied	PHP	08	1	------
PLA Pull accumulator from stack	A ↑	Implied	PLA	68	1	√√----
PLP Pull processor status from stack	P ↑	Implied	PLP	28	1	From Stack
ROL Rotate one bit left (memory or accumulator)	(See Figure 2)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	ROL A ROL Oper ROL Oper,X ROL Oper ROL Oper,X	2A 26 36 2E 3E	1 2 2 3 3	√√√---
ROR Rotate one bit right (memory or accumulator)	(See Figure 3)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	ROR A ROR Oper ROR Oper,X ROR Oper ROR Oper,X	6A 66 76 6E 7E	1 2 2 3 3	√√√---

Nimi Kuvaus	Toiminto	Osoitus- muoto	Assembly- kielinen muoto	HEX OP- koodi	Tilan- tarve byteä	»P» status rekisteri N Z C I D V
RTI Return from interrupt	P↑ PC↑	Implied	RTI	40	1	From Stack
RTS Return from subroutine	PC↑, PC+1 → PC	Implied	RTS	60	1	------
SBC Subtract memory from accumulator with borrow	A - M - $\overline{C}$ → A	Immediate	SBC #Oper	E9	2	√√√---√
		Zero Page	SBC Oper	E5	2	
		Zero Page,X	SBC Oper,X	F5	2	
		Absolute	SBC Oper	ED	3	
		Absolute,X	SBC Oper,X	FD	3	
		Absolute,Y	SBC Oper,Y	F9	3	
		(Indirect,X)	SBC (Oper,X)	E1	2	
		(Indirect),Y	SBC (Oper),Y	F1	2	
SEC Set carry flag	1 → C	Implied	SEC	38	1	--1---
SED Set decimal mode	1 → D	Implied	SED	F8	1	----1-
SEI Set interrupt disable status	1 → I	Implied	SEI	78	1	---1--
STA Store accumulator in memory	A → M	Zero Page	STA Oper	85	2	------
		Zero Page,X	STA Oper,X	95	2	
		Absolute	STA Oper	8D	3	
		Absolute,X	STA Oper,X	9D	3	
		Absolute,Y	STA Oper,Y	99	3	
		(Indirect,X)	STA (Oper,X)	81	2	
		(Indirect),Y	STA (Oper),Y	91	2	
STX Store index X in memory	X → M	Zero Page	STX Oper	86	2	------
		Zero Page,Y	STX Oper,Y	96	2	
		Absolute	STX Oper	8E	3	
STY Store index Y in memory	Y → M	Zero Page	STY Oper	84	2	------
		Zero Page,X	STY Oper,X	94	2	
		Absolute	STY Oper	8C	3	
TAX Transfer accumulator to index X	A → X	Implied	TAX	AA	1	√√----
TAY Transfer accumulator to index Y	A → Y	Implied	TAY	A8	1	√√----
TSX Transfer stack pointer to index X	S → X	Implied	TSX	BA	1	√√----

Nimi Kuvaus	Toiminto	Osoitus-muoto	Assembly-kielinen muoto	HEX OP-koodi	Tilan-tarve bytea	»P» status rekisteri N Z C I D V
TXA Transfer index X to accumulator	X → A	Implied	TXA	8A	1	√√----
TXS Transfer index X to stack pointer	X → S	Implied	TXS	9A	1	------
TYA Transfer index Y to accumulator	Y → A	Implied	TYA	98	1	√√----

KÄSKYT HEXADESIMAALISINA

00 — BRK
01 — ORA — (Indirect, X)
02 — NOP
03 — NOP
04 — NOP
05 — ORA — Zero Page
06 — ASL — Zero Page
07 — NOP
08 — PHP
09 — ORA — Immediate
0A — ASL — Accumulator
0B — NOP
0C — NOP
0D — ORA — Absolute
0E — ASL — Absolute
0F — NOP
10 — BPL
11 — ORA — (Indirect), Y
12 — NOP
13 — NOP
14 — NOP
15 — ORA — Zero Page, X
16 — ASL — Zero Page, X
17 — NOP
18 — CLC
19 — ORA — Absolute, Y
1A — NOP
1B — NOP
1C — NOP
1D — ORA — Absolute, X
1E — ASL — Absolute, X
1F — NOP
20 — JSR
21 — AND — (Indirect, X)
22 — NOP
23 — NOP
24 — BIT — Zero Page
25 — AND — Zero Page
26 — ROL — Zero Page
27 — NOP
28 — PLP
29 — AND — Immediate
2A — ROL — Accumulator
2B — NOP
2C — BIT — Absolute
2D — AND — Absolute
2E — ROL — Absolute
2F — NOP
30 — BMI
31 — AND — (Indirect), Y
32 — NOP
33 — NOP
34 — NOP
35 — AND — Zero Page, X
36 — ROL — Zero Page, X
37 — NOP
38 — SEC
39 — AND — Absolute, Y
3A — NOP
3B — NOP
3C — NOP
3D — AND — Absolute, X
3E — ROL — Absolute, X
3F — NOP
40 — RTI
41 — EOR — (Indirect, X)
42 — NOP
43 — NOP
44 — NOP
45 — EOR — Zero Page
46 — LSR — Zero Page
47 — NOP
48 — PHA
49 — EOR — Immediate
4A — LSR — Accumulator
4B — NOP
4C — JMP — Absolute
4D — EOR — Absolute
4E — LSR — Absolute
4F — NOP
50 — BVC
51 — EOR (Indirect), Y
52 — NOP
53 — NOP
54 — NOP
55 — EOR — Zero Page, X
56 — LSR — Zero Page, X
57 — NOP
58 — CLI
59 — EOR — Absolute, Y
5A — NOP
5B — NOP
5C — NOP
5D — EOR — Absolute, X
5E — LSR — Absolute, X
5F — NOP
60 — RTS
61 — ADC — (Indirect, X)
62 — NOP
63 — NOP
64 — NOP
65 — ADC — Zero Page
66 — ROR — Zero Page
67 — NOP
68 — PLA
69 — ADC — Immediate
6A — ROR — Accumulator
6B — NOP
6C — JMP — Indirect
6D — ADC — Absolute
6E — ROR — Absolute
6F — NOP
70 — BVS
71 — ADC — (Indirect), Y
72 — NOP
73 — NOP
74 — NOP
75 — ADC — Zero Page, X
76 — ROR — Zero Page, X
77 — NOP
78 — SEI
79 — ADC — Absolute, Y
7A — NOP
7B — NOP
7C — NOP
7D — ADC — Absolute, X NOP
7E — ROR — Absolute, X NOP
7F — NOP
80 — NOP
81 — STA — (Indirect, X)
82 — NOP
83 — NOP
84 —STY — Zero Page
85 — STA — Zero Page
86 — STX — Zero Page
87 — NOP
88 — DEY
89 — NOP
8A — TXA
8B — NOP
8C — STY — Absolute

8D — STA — Absolute
8E — STX — Absolute
8F — NOP
90 — BCC
91 — STA — (Indirect), Y
92 — NOP
93 — NOP
94 — STY — Zero Page, X
95 — STA — Zero Page, X
96 — STX — Zero Page, Y
97 — NOP
98 — TYA
99 — STA — Absolute, Y
9A — TXS
9B — NOP
9C — NOP
9D — STA — Absolute, X
9E — NOP
9F — NOP
A0 — LDY — Immediate
A1 — LDA — (Indirect, X)
A2 — LDX — Immediate
A3 — NOP
A4 — LDY — Zero Page
A5 — LDA — Zero Page
A6 — LDX — Zero Page
A7 — NOP
A8 — TAY
A9 — LDA — Immediate
AA — TAX
AB — NOP
AC — LDY — Absolute
AD — Absolute
AE — LDX — Absolute
AF — NOP
B0 — BCS
B1 — LDA — (Indirect), Y
B2 — NOP
B3 — NOP
B4 — LDY — Zero Page, X
B5 — LDA — Zero Page, X
B6 — LDX — Zero Page, Y
B7 — NOP
B8 — CLV
B9 — LDA — Absolute, Y
BA — TSX
BB — NOP
BC — LDY — Absolute, X
BD — LDA — Absolute, X
BE — LDX — Absolute, Y
BF — NOP
C0 — CPY — Immediate
C1 — CMP — (Indirect, X)
C2 — NOP
C3 — NOP
C4 — CPY — Zero Page
C5 — CMP — Zero Page
C6 — DEC — Zero Page
C7 — NOP
C8 — INY
C9 — CMP — Immediate
CA — DEX
CB — NOP
CC — CPY — Absolute
CD — CMP — Absolute
CE — DEC — Absolute
CF — NOP
D0 — BNE
D1 — CMP — (Indirect), Y
D2 — NOP
D3 — NOP
D4 — NOP
D5 — CMP — Zero Page, X
D6 — DEC — Zero Page, X
D7 — NOP
D8 — CLD
D9 — CMP — Absolute, Y
DA — NOP
DB — NOP
DC — NOP
DD — CMP — Absolute, X
DE — DEC — Absolute, X
DF — NOP
E0 — CPX — Immediate
E1 — SBC — (Indirect, X)
E2 — NOP
E3 — NOP
E4 — CPX — Zero Page
E5 — SBC — Zero Page
E6 — INC — Zero Page
E7 — NOP
E8 — INX
E9 — SBC — Immediate
EA — NOP
EB — NOP
EC — CPX — Absolute
ED — SBC — Absolute
EE — INC — Absolute
EF — NOP
F0 — BEQ
F1 — SBC — (Indirect), Y
F2 — NOP
F3 — NOP
F4 — NOP
F5 — SBC — Zero Page, X
F6 — INC — Zero Page, X
F7 — NOP
F8 — SED
F9 — SBC — Absolute, Y
FA — NOP
FB — NOP
FC — NOP
FD — SBC — Absolute, X
FE — INC — Absolute, X
FF — NOP

LIITE B
ERIKOISMUISTIPAIKAT

Taulukko 1: Näppäimistön erikoismuistipaikat			
Muistipaikka Hex	Desim.		Kuvaus
$C000	49152	-16384	Näppäimistön data
$C010	49168	-16368	Näppäimistön Stroben nollaus.

Taulukko 4: Videonäytön käyttämät muistialueet					
Näyttö	Sivu	Alkukohta Hexa	Desimaali	Loppukohta Hex	Decimal
Teksti/ perusgrafiikka	Sivu 1	$400	1024	$7FF	2047
	Sivu 2	$800	2048	$BFF	3071
Tarkkuus-grafiikka	Sivu 1	$2000	8192	$3FFF	16383
	Sivu 2	$4000	16384	$5FFF	24575

Taulukko 5: Näytön ohjelmalliset kytkimet			
Muistipaikka Hex	Desim.		Kuvaus
$C050	49232	-16304	Näyttöön jokin Grafiikkatila.
$C051	49233	-16303	Näyttö tekstille.
$C052	49234	-16302	Näyttö kokonaan tekstille tai grafiikalle.
$C053	49235	-16301	Sekoitettu teksti ja grafiikka.
$C054	49236	-16300	Näytä perussivu (sivu 1)
$C055	49237	-16299	Näytä apusivu (sivu 2)
$C056	49238	-16298	Näyttöön perusgrafiikka.
$C057	49239	-16297	Näyttöön tarkkuusgrafiikka.

Taulukko 9: Kutsu-ulostulojen osoitteet				
	Tila	Osoite: Desim.		Hex
0	ei	49240	-16296	$C058
	on	49241	-16295	$C059
1	ei	49242	-16294	$C05A
	on	49243	-16293	$C05B
2	ei	49244	-16292	$C05C
	on	49245	-16291	$C05D
3	ei	49246	-16290	$C05E
	on	49247	-16289	$C05F

Taulukko 10: I/O:n erikoismuistipaikat

Toiminta	Osoite: Desim.		Hexa	Luku / Kirjoitus
Kaiutin	49200	-16336	$C030	L
Kasetti-sisaanmeno	49184	-16352	$C020	L
Kasetti-ulostulo	49256	-16288	$C060	L
Kutsu-ulostulot	49240	-16296	$C058	L/K
	...	...	...	
	49247	-16289	$C05F	
Lippu-sisäänmenot	49249	-16287	$C061	L
	49250	-16286	$C062	L
	49251	-16285	$C063	L
Analogia-sisäänmenot	49252	-16284	$C064	L
	49253	-16283	$C065	
	49254	-16282	$C066	
	49255	-16281	$C067	
Analogia-nollaus	49264	-16272	$C070	L/K
Utility Strobe	49216	-16320	$C040	L

Taulukko 11: Teksti-ikkunan erikoismuistipaikat

Toiminta	Muistipaikka Desim.	Hex	Minimi/Normaali/Maksimiarvo Desim.	Hex
Vasen reuna	32	$20	0/0/39	$0/$0/$17
Leveys	33	$21	0/40/40	$0/$28/$28
Yläreuna	34	$22	0/0/24	$0/$0/$18
Alareuna	35	$23	0/24/24	$0/$18/$18

Taulukko 12: Normaali/käänteinen-tilan ohjausarvot

Arvo: Desim.	Hex	Vaikutus:
255	$FF	COUT tulostaa merkit normaalitilassa.
63	$3F	COUT tulostaa merkit käänteisessä tekstitilassa.
127	$7F	COUT tulostaa kirjaimet vilkkuvina, muut merkit käänteisinä.

Taulukko 13. Autostart ROMin erikoismuistipaikat

Muistipaikka Desim.	Hex	Sisältö:
1010 1011	$3F2 $3F3	Paluuvektori (Soft entry vector). Muistipaikat sisältävät paluuosoitteen kulloin käytössä olevaan kieleen. Tavallisesti $E003.
1012	$3F4	Käynnistystavu. Normaali sisältö $45. Katso alla.
64367 (-1169)	$FB6F	Alkuosoite konekieliseen ohjelmaan, joka asettaa käynnistystavun.

Taulukko 14: Sivun kolme Monitorimuistipaikat			
Osoite: Desim.	 Hex	Käyttö: Monitori ROM	 Autostart ROM
1008 1009	$3F0 $3F1	Ei käytössä.	Pitää sisällään muistipaikan osoitteen, joka käsittelee konekielisen keskeytyspyyntötoiminnon (BRK-request) (tavallisesti $FA59)
1010 1011	$3F2 $3F3	Ei käytössä.	Paluu (soft entry) vektori
1012	$3F4	Ei käytössä.	Käynnistystavu
1013 1014 1015	$3F5 $3F6 $3F7	Sisältää hyppykäskyn (JMP) aliohjelmaan, joka käsittelee Applesoft II:n »8»-käskyjä . Tavallisesti $4C $58 $FF.	
1016 1017 1018	$3F8 $3F9 $3FA	Sisältää hyppykäskyn (JMP) aliohjelmaan, joka käsittelee monitorin (CTRL Y) -käskyt.	
1019 1020 1021	$3FB $3FC $3FD	Sisältää hyppykäskyn (JMP) aliohjelmaan, joka käsittelee NMI-keskeytyspyynnöt (Nom Maskable Interrupts).	
1022 1023	$3FE $3FF	Sisältää osoitteen aliohjelmaan, joka käsittelee IRQ-keskeytyspyynnöt (Interrupt Requests).	

Taulukko 22: Sisäänrakennetut I/O-muistipaikat

	$0	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
$C000	Näppäimistön data-otto															
$C010	Näppäimistön strobennollaus															
$C020	Kasettiliitännän ulostulon sysäys															
$C030	Kaiuttimen sysäys															
$C040	Utility-strobe															
$C050	gr	tx	nomix	mix	pri	sec	lores	hires	an0		an1		an2		an3	
$C060	cin	pb1	pb2	pb3	gc0	gc1	gc2	gc3	toisto $C060-$C067							
$C070	Game Controller Strobe															

Peliohjainten Strobe:

gr	Aseta grafiikkatila	tx	Aseta tekstitila
nomix	Aseta teksti/tai grafiikkatila (kokonäyttö)	mix	Sekoitettu teksti/grafiikkatila
pri	Näytä perussivu (sivu 1)	sec	Näytä varasivu (sivu 2)
lores	Näytä perusgrafiikka	hires	Näytä tarkkuusgrafiikka
an	Ulostulot	pb	Näppäinsisäänmenot (peliohjain)
gc	Peliohjainten sisäänmenot	cin	Kasettisisäänmeno

Taulukko 23: Oheislaitekorttien I/O-muistipaikat

	$Ø	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
$CØ8Ø										Ø						
$CØ9Ø										1						
$CØAØ										2						
$CØBØ				Otto/anto kullekin väylänumerolle						3						
$CØCØ										4						
$CØDØ										5						
$CØEØ										6						
$CØFØ										7						

Taulukko 24: Oheislaitekorttien PROM-muistipaikat

	$ØØ	$1Ø	$2Ø	$3Ø	$4Ø	$5Ø	$6Ø	$7Ø	$8Ø	$9Ø	$AØ	$BØ	$CØ	$DØ	$EØ	$FØ
$C1ØØ										1						
$C2ØØ										2						
$C3ØØ										3						
$C4ØØ				PROM-tila kullekin väylänumerolle						4						
$C5ØØ										5						
$C6ØØ										6						
$C7ØØ										7						

Taulukko 25: I/O-muistipaikkojen perusosoitteet

Perus-osoite	Väylä Ø	1	2	3	4	5	6	7
$CØ8Ø	$CØ8Ø	$CØ9Ø	$CØAØ	$CØBØ	$CØCØ	$CØDØ	$CØEØ	$CØFØ
$CØ81	$CØ81	$CØ91	$CØA1	$CØB1	$CØC1	$CØD1	$CØE1	$CØF1
$CØ82	$CØ82	$CØ92	$CØA2	$CØB2	$CØC2	$CØD2	$CØE2	$CØF2
$CØ83	$CØ83	$CØ93	$CØA3	$CØB3	$CØC3	$CØD3	$CØE3	$CØF3
$CØ84	$CØ84	$CØ94	$CØA4	$CØB4	$CØC4	$CØD4	$CØF4	$CØF4
$CØ85	$CØ85	$CØ95	$CØA5	$CØB5	$CØC5	$CØD5	$CØE5	$CØF5
$CØ86	$CØ86	$CØ96	$CØA6	$CØB6	$CØC6	$CØD6	$CØE6	$CØF6
$CØ87	$CØ87	$CØ97	$CØA7	$CØB7	$CØC7	$CØD7	$CØE7	$CØF7
$CØ88	$CØ88	$CØ98	$CØA8	$CØB8	$CØC8	$CØD8	$CØE8	$CØF8
$CØ89	$CØ89	$CØ99	$CØA9	$CØB9	$CØC9	$CØD9	$CØE9	$CØF9
$CØ8A	$CØ8A	$CØ9A	$CØAA	$CØBA	$CØCA	$CØDA	$CØEA	$CØFA
$CØ8B	$CØ8B	$CØ9B	$CØAB	$CØBB	$CØCB	$CØDB	$CØEB	$CØFB
$CØ8C	$CØ8C	$CØ9C	$CØAC	$CØBC	$CØCC	$CØDC	$CØEC	$CØFC
$CØ8D	$CØ8D	$CØ9D	$CØAD	$CØBD	$CØCD	$CØDD	$CØED	$CØFD
$CØ8E	$CØ8E	$CØ9E	$CØAE	$CØBE	$CØCE	$CØDE	$CØEE	$CØFE
$CØ8F	$CØ8F	$CØ9F	$CØAF	$CØBF	$CØCF	$CØDF	$CØEF	$CØFF
	I/O-muistipaikat							

Taulukko 26. I/O-muistialueen RAM-osoitteet (scratchpad)							
Perus-osoite	Väylänumero						
	1	2	3	4	5	6	7
$Ø478	$Ø479	$Ø47A	$Ø47B	$Ø47C	$Ø47D	$Ø47E	$Ø47F
$Ø4F8	$Ø4F9	$Ø4FA	$Ø4FB	$Ø4FC	$Ø4FD	$Ø4FE	$Ø4FF
$Ø578	$Ø579	$Ø57A	$Ø57B	$Ø57C	$Ø57D	$Ø57E	$Ø57F
$Ø5F8	$Ø5F9	$Ø5FA	$Ø5FB	$Ø5FC	$Ø5FD	$Ø5FE	$Ø5FF
$Ø678	$Ø679	$Ø67A	$Ø67B	$Ø67C	$Ø67D	$Ø67E	$Ø67F
$Ø6F8	$Ø6F9	$Ø6FA	$Ø6FB	$Ø6FC	$Ø6FD	$Ø6FE	$Ø6FF
$Ø778	$Ø779	$Ø77A	$Ø77B	$Ø77C	$Ø77D	$Ø77E	$Ø77F
$Ø7F8	$Ø7F9	$Ø7FA	$Ø7FB	$Ø7FC	$Ø7FD	$Ø7FE	$Ø7FF

LIITE C
ROM · LISTAUKSET

AUTOSTART ROMin LISTAUS

```
0000:          ********************************
0000:          *
0000:          * APPLE II
0000:          * MONITOR II
0000:          *
0000:          * COPYRIGHT 1978 BY
0000:          * APPLE COMPUTER, INC.
0000:          *
0000:          * ALL RIGHTS RESERVED
0000:          *
0000:          * STEVE WOZNIAK
0000:          *
0000:          ********************************
0000:          *
0000:          * MODIFIED NOV 1978
0000:          * BY JOHN A
0000:          *
0000:          ********************************
F800:                   ORG $F800
F800:                   OBJ $2000
F800:          ******************************
F800:          LOCO     EQU $00
F800:          LOC1     EQU $01
F800:          WNDLFT   EQU $20
F800:          WNDWDTH  EQU $21
F800:          WNDTOP   EQU $22
F800:          WNDBTM   EQU $23
F800:          CH       EQU $24
F800:          CV       EQU $25
F800:          GBASL    EQU $26
F800:          GBASH    EQU $27
F800:          BASL     EQU $28
F800:          BASH     EQU $29
F800:          BAS2L    EQU $2A
F800:          BAS2H    EQU $2B
F800:          H2       EQU $2C
F800:          LMNEM    EQU $2C
F800:          V2       EQU $2D
F800:          RMNEM    EQU $2D
F800:          MASK     EQU $2E
F800:          CHKSUM   EQU $2E
F800:          FORMAT   EQU $2E
F800:          LASTIN   EQU $2F
F800:          LENGTH   EQU $2F
F800:          SIGN     EQU $2F
F800:          COLOR    EQU $30
F800:          MODE     EQU $31
F800:          INVFLG   EQU $32
F800:          PROMPT   EQU $33
F800:          YSAV     EQU $34
F800:          YSAV1    EQU $35
F800:          CSWL     EQU $36
F800:          CSWH     EQU $37
F800:          KSWL     EQU $38
F800:          KSWH     EQU $39
F800:          PCL      EQU $3A
F800:          PCH      EQU $3B
F800:          A1L      EQU $3C
F800:          A1H      EQU $3D
F800:          A2L      EQU $3E
F800:          A2H      EQU $3F
F800:          A3L      EQU $40
F800:          A3H      EQU $41
F800:          A4L      EQU $42
F800:          A4H      EQU $43
F800:          A5L      EQU $44
F800:          A5H      EQU $45
```

```
F800:            ACC      EQU $45      ; NOTE OVERLAP WITH A5H!
F800:            XREG     EQU $46
F800:            YREG     EQU $47
F800:            STATUS   EQU $48
F800:            SPNT     EQU $49
F800:            RNDL     EQU $4E
F800:            RNDH     EQU $4F
F800:            PICK     EQU $95
F800:            IN       EQU $0200
F800:            BRKV     EQU $3F0     ; NEW VECTOR FOR BRK
F800:            SOFTEV   EQU $3F2     ; VECTOR FOR WARM START
F800:            PWREDUP  EQU $3F4     ; THIS MUST = EOR #$A5 OF SOFTEV+1
F800:            AMPERV   EQU $3F5     ; APPLESOFT & EXIT VECTOR
F800:            USRADR   EQU $03F8
F800:            NMI      EQU $03FB
F800:            IRQLOC   EQU $3FE
F800:            LINE1    EQU $400
F800:            MSLOT    EQU $07F8
F800:            IOADR    EQU $C000
F800:            KBD      EQU $C000
F800:            KBDSTRB  EQU $C010
F800:            TAPEOUT  EQU $C020
F800:            SPKR     EQU $C030
F800:            TXTCLR   EQU $C050
F800:            TXTSET   EQU $C051
F800:            MIXCLR   EQU $C052
F800:            MIXSET   EQU $C053
F800:            LOWSCR   EQU $C054
F800:            HISCR    EQU $C055
F800:            LORES    EQU $C056
F800:            HIRES    EQU $C057
F800:            SETAN0   EQU $C058
F800:            CLRAN0   EQU $C059
F800:            SETAN1   EQU $C05A
F800:            CLRAN1   EQU $C05B
F800:            SETAN2   EQU $C05C
F800:            CLRAN2   EQU $C05D
F800:            SETAN3   EQU $C05E
F800:            CLRAN3   EQU $C05F
F800:            TAPEIN   EQU $C060
F800:            PADDL0   EQU $C064
F800:            PTRIG    EQU $C070
F800:            CLRROM   EQU $CFFF
F800:            BASIC    EQU $E000
F800:            BASIC2   EQU $E003
F800:                     PAGE
F800: 4A         PLOT     LSR A
F801: 08                  PHP
F802: 20 47 F8            JSR GBASCALC
F805: 28                  PLP
F806: A9 0F               LDA #$0F
F808: 90 02               BCC RTMASK
F80A: 69 E0               ADC #$E0
F80C: 85 2E      RTMASK   STA MASK
F80E: B1 26      PLOT1    LDA (GBASL),Y
F810: 45 30               EOR COLOR
F812: 25 2E               AND MASK
F814: 51 26               EOR (GBASL),Y
F816: 91 26               STA (GBASL),Y
F818: 60                  RTS
F819: 20 00 F8   HLINE    JSR PLOT
F81C: C4 2C      HLINE1   CPY H2
F81E: B0 11               BCS RTS1
F820: C8                  INY
F821: 20 0E F8            JSR PLOT1
F824: 90 F6               BCC HLINE1
F826: 69 01      VLINEZ   ADC #$01
F828: 48         VLINE    PHA
F829: 20 00 F8            JSR PLOT
F82C: 68                  PLA
F82D: C5 2D               CMP V2
F82F: 90 F5               BCC VLINEZ
F831: 60         RTS1     RTS
```

```
F832:  A0 2F       CLRSCR   LDY #$2F
F834:  D0 02                BNE CLRSC2
F836:  A0 27       CLRTOP   LDY #$27
F838:  84 2D       CLRSC2   STY V2
F83A:  A0 27                LDY #$27
F83C:  A9 00       CLRSC3   LDA #$00
F83E:  85 30                STA COLOR
F840:  20 28 F8             JSR VLINE
F843:  88                   DEY
F844:  10 F6                BPL CLRSC3
F846:  60                   RTS
F847:                       PAGE
F847:  48          GBASCALC PHA
F848:  4A                   LSR A
F849:  29 03                AND #$03
F84B:  09 04                ORA #$04
F84D:  85 27                STA GBASH
F84F:  68                   PLA
F850:  29 18                AND #$18
F852:  90 02                BCC GBCALC
F854:  69 7F                ADC #$7F
F856:  85 26       GBCALC   STA GBASL
F858:  0A                   ASL A
F859:  0A                   ASL A
F85A:  05 26                ORA GBASL
F85C:  85 26                STA GBASL
F85E:  60                   RTS
F85F:  A5 30                LDA COLOR
F861:  18                   CLC
F862:  69 03                ADC #$03
F864:  29 0F       SETCOL   AND #$0F
F866:  85 30                STA COLOR
F868:  0A                   ASL A
F869:  0A                   ASL A
F86A:  0A                   ASL A
F86B:  0A                   ASL A
F86C:  05 30                ORA COLOR
F86E:  85 30                STA COLOR
F870:  60                   RTS
F871:  4A          SCRN     LSR A
F872:  08                   PHP
F873:  20 47 F8             JSR GBASCALC
F876:  B1 26                LDA (GBASL),Y
F878:  28                   PLP
F879:  90 04       SCRN2    BCC RTMSKZ
F87B:  4A                   LSR A
F87C:  4A                   LSR A
F87D:  4A                   LSR A
F87E:  4A                   LSR A
F87F:  29 0F       RTMSKZ   AND #$0F
F881:  60                   RTS
F882:                       PAGE
F882:  A6 3A       INSDS1   LDX PCL
F884:  A4 3B                LDY PCH
F886:  20 96 FD             JSR PRYX2
F889:  20 48 F9             JSR PRBLNK
F88C:  A1 3A       INSDS2   LDA (PCL,X)
F88E:  A8                   TAY
F88F:  4A                   LSR A
F890:  90 09                BCC IEVEN
F892:  6A                   ROR A
F893:  B0 10                BCS ERR
F895:  C9 A2                CMP #$A2
F897:  F0 0C                BEQ ERR
F899:  29 87                AND #$87
F89B:  4A          IEVEN    LSR A
F89C:  AA                   TAX
F89D:  BD 62 F9             LDA FMT1,X
F8A0:  20 79 F8             JSR SCRN2
F8A3:  D0 04                BNE GETFMT
F8A5:  A0 80       ERR      LDY #$80
F8A7:  A9 00                LDA #$00
F8A9:  AA          GETFMT   TAX
```

```
F8AA:  BD A6 F9            LDA FMT2,X
F8AD:  85 2E               STA FORMAT
F8AF:  29 03               AND #$03
F8B1:  85 2F               STA LENGTH
F8B3:  98                  TYA
F8B4:  29 8F               AND #$8F
F8B6:  AA                  TAX
F8B7:  98                  TYA
F8B8:  A0 03               LDY #$03
F8BA:  E0 8A               CPX #$8A
F8BC:  F0 0B               BEQ MNNDX3
F8BE:  4A        MNNDX1    LSR A
F8BF:  90 08               BCC MNNDX3
F8C1:  4A                  LSR A
F8C2:  4A        MNNDX2    LSR A
F8C3:  09 20               ORA #$20
F8C5:  88                  DEY
F8C6:  D0 FA               BNE MNNDX2
F8C8:  C8                  INY
F8C9:  88        MNNDX3    DEY
F8CA:  D0 F2               BNE MNNDX1
F8CC:  60                  RTS
F8CD:  FF FF FF            DFB $FF,$FF,$FF
F8D0:                      PAGE
F8D0:  20 82 F8  INSTDSP   JSR INSDS1
F8D3:  48                  PHA
F8D4:  B1 3A     PRNTOP    LDA (PCL),Y
F8D6:  20 DA FD            JSR PRBYTE
F8D9:  A2 01               LDX #$01
F8DB:  20 4A F9  PRNTBL    JSR PRBL2
F8DE:  C4 2F               CPY LENGTH
F8E0:  C8                  INY
F8E1:  90 F1               BCC PRNTOP
F8E3:  A2 03               LDX #$03
F8E5:  C0 04               CPY #$04
F8E7:  90 F2               BCC PRNTBL
F8E9:  68                  PLA
F8EA:  A8                  TAY
F8EB:  B9 C0 F9            LDA MNEML,Y
F9EE:  85 2C               STA LMNEM
F8F0:  B9 00 FA            LDA MNEMR,Y
F8F3:  85 2D               STA RMNEM
F8F5:  A9 00     NXTCOL    LDA #$00
F8F7:  A0 05               LDY #$05
F8F9:  06 2D     PRMN2     ASL RMNEM
F8FB:  26 2C               ROL LMNEM
F8FD:  2A                  ROL A
F8FE:  88                  DEY
F8FF:  D0 F8               BNE PRMN2
F901:  69 BF               ADC #$BF
F903:  20 ED FD            JSR COUT
F906:  CA                  DEX
F907:  D0 EC               BNE NXTCOL
F909:  20 48 F9            JSR PRBLNK
F90C:  A4 2F               LDY LENGTH
F90E:  A2 06               LDX #$06
F910:  E0 03     PRADR1    CPX #$03
F912:  F0 1C               BEQ PRADR5
F914:  06 2E     PRADR2    ASL FORMAT
F916:  90 0E               BCC PRADR3
F918:  BD B3 F9            LDA CHAR1-1,X
F91B:  20 ED FD            JSR COUT
F91E:  BD B9 F9            LDA CHAR2-1,X
F921:  F0 03               BEQ PRADR3
F923:  20 ED FD            JSR COUT
F926:  CA        PRADR3    DEX
F927:  D0 E7               BNE PRADR1
F929:  60                  RTS
F92A:  88        PRADR4    DEY
F92B:  30 E7               BMI PRADR2
F92D:  20 DA FD            JSR PRBYTE
F930:  A5 2E     PRADR5    LDA FORMAT
F932:  C9 E8               CMP #$E8
```

```
F934: B1 3A              LDA (PCL),Y
F936: 90 F2              BCC PRADR4
F938:                    PAGE
F938: 20 56 F9    RELADR JSR PCADJ3
F93B: AA                 TAX
F93C: E8                 INX
F93D: D0 01              BNE PRNTYX
F93F: C8                 INY
F940: 98          PRNTYX TYA
F941: 20 DA FD    PRNTAX JSR PRBYTE
F944: 8A          PRNTX  TXA
F945: 4C DA FD           JMP PRBYTE
F948: A2 03       PRBLNK LDX #$03
F94A: A9 A0       PRBL2  LDA #$A0
F94C: 20 ED FD    PRBL3  JSR COUT
F94F: CA                 DEX
F950: D0 F8              BNE PRBL2
F952: 60                 RTS
F953: 38          PCADJ  SEC
F954: A5 2F       PCADJ2 LDA LENGTH
F956: A4 3B       PCADJ3 LDY PCH
F958: AA                 TAX
F959: 10 01              BPL PCADJ4
F95B: 88                 DEY
F95C: 65 3A       PCADJ4 ADC PCL
F95E: 90 01              BCC RTS2
F960: C8                 INY
F961: 60          RTS2   RTS
F962: 04          FMT1   DFB $04
F963: 20                 DFB $20
F964: 54                 DFB $54
F965: 30                 DFB $30
F966: 0D                 DFB $0D
F967: 80                 DFB $80
F968: 04                 DFB $04
F969: 90                 DFB $90
F96A: 03                 DFB $03
F96B: 22                 DFB $22
F96C: 54                 DFB $54
F96D: 33                 DFB $33
F96E: 0D                 DFB $0D
F96F: 80                 DFB $80
F970: 04                 DFB $04
F971: 90                 DFB $90
F972: 04                 DFB $04
F973: 20                 DFB $20
F974: 54                 DFB $54
F975: 33                 DFB $33
F976: 0D                 DFB $0D
F977: 80                 DFB $80
F978: 04                 DFB $04
F979: 90                 DFB $90
F97A: 04                 DFB $04
F97B: 20                 DFB $20
F97C: 54                 DFB $54
F97D: 3B                 DFB $3B
F97E: 0D                 DFB $0D
F97F: 80                 DFB $80
F980: 04                 DFB $04
F981: 90                 DFB $90
F982: 00                 DFB $00
F983: 22                 DFB $22
F984: 44                 DFB $44
F985: 33                 DFB $33
F986: 0D                 DFB $0D
F987: C8                 DFB $C8
F988: 44                 DFB $44
F989: 00                 DFB $00
F98A: 11                 DFB $11
F98B: 22                 DFB $22
F98C: 44                 DFB $44
F98D: 33                 DFB $33
F98E: 0D                 DFB $0D
```

```
F98F:  C8                 DFB $C8
F990:  44                 DFB $44
F991:  A9                 DFB $A9
F992:  01                 DFB $01
F993:  22                 DFB $22
F994:  44                 DFB $44
F995:  33                 DFB $33
F996:  0D                 DFB $0D
F997:  80                 DFB $80
F998:  04                 DFB $04
F999:  90                 DFB $90
F99A:  01                 DFB $01
F99B:  22                 DFB $22
F99C:  44                 DFB $44
F99D:  33                 DFB $33
F99E:  0D                 DFB $0D
F99F:  80                 DFB $80
F9A0:  04                 DFB $04
F9A1:  90                 DFB $90
F9A2:  26                 DFB $26
F9A3:  31                 DFB $31
F9A4:  87                 DFB $87
F9A5:  9A                 DFB $9A
F9A6:  00        FMT2     DFB $00
F9A7:  21                 DFB $21
F9A8:  81                 DFB $81
F9A9:  82                 DFB $82
F9AA:  00                 DFB $00
F9AB:  00                 DFB $00
F9AC:  59                 DFB $59
F9AD:  4D                 DFB $4D
F9AE:  91                 DFB $91
F9AF:  92                 DFB $92
F9B0:  86                 DFB $86
F9B1:  4A                 DFB $4A
F9B2:  85                 DFB $85
F9B3:  9D                 DFB $9D
F9B4:  AC        CHAR1    DFB $AC
F9B5:  A9                 DFB $A9
F9B6:  AC                 DFB $AC
F9B7:  A3                 DFB $A3
F9B8:  A8                 DFB $A8
F9B9:  A4                 DFB $A4
F9BA:  D9        CHAR2    DFB $D9
F9BB:  00                 DFB $00
F9BC:  D8                 DFB $D8
F9BD:  A4                 DFB $A4
F9BE:  A4                 DFB $A4
F9BF:  00                 DFB $00
F9C0:  1C        MNEML    DFB $1C
F9C1:  8A                 DFB $8A
F9C2:  1C                 DFB $1C
F9C3:  23                 DFB $23
F9C4:  5D                 DFB $5D
F9C5:  8B                 DFB $8B
F9C6:  1B                 DFB $1B
F9C7:  A1                 DFB $A1
F9C8:  9D                 DFB $9D
F9C9:  8A                 DFB $8A
F9CA:  1D                 DFB $1D
F9CB:  23                 DFB $23
F9CC:  9D                 DFB $9D
F9CD:  8B                 DFB $8B
F9CE:  1D                 DFB $1D
F9CF:  A1                 DFB $A1
F9D0:  00                 DFB $00
F9D1:  29                 DFB $29
F9D2:  19                 DFB $19
F9D3:  AE                 DFB $AE
F9D4:  69                 DFB $69
F9D5:  A8                 DFB $A8
F9D6:  19                 DFB $19
F9D7:  23                 DFB $23
```

```
F9D8:  24                      DFB $24
F9D9:  53                      DFB $53
F9DA:  1B                      DFB $1B
F9DB:  23                      DFB $23
F9DC:  24                      DFB $24
F9DD:  53                      DFB $53
F9DE:  19                      DFB $19
F9DF:  A1                      DFB $A1
F9E0:  00                      DFB $00
F9E1:  1A                      DFB $1A
F9E2:  5B                      DFB $5B
F9E3:  5B                      DFB $5B
F9E4:  A5                      DFB $A5
F9E5:  69                      DFB $69
F9E6:  24                      DFB $24
F9E7:  24                      DFB $24
F9E8:  AE                      DFB $AE
F9E9:  AE                      DFB $AE
F9EA:  A8                      DFB $A8
F9EB:  AD                      DFB $AD
F9EC:  29                      DFB $29
F9ED:  00                      DFB $00
F9EE:  7C                      DFB $7C
F9EF:  00                      DFB $00
F9F0:  15                      DFB $15
F9F1:  9C                      DFB $9C
F9F2:  6D                      DFB $6D
F9F3:  9C                      DFB $9C
F9F4:  A5                      DFB $A5
F9F5:  69                      DFB $69
F9F6:  29                      DFB $29
F9F7:  53                      DFB $53
F9F8:  84                      DFB $84
F9F9:  13                      DFB $13
F9FA:  34                      DFB $34
F9FB:  11                      DFB $11
F9FC:  A5                      DFB $A5
F9FD:  69                      DFB $69
F9FE:  23                      DFB $23
F9FF:  A0                      DFB $A0
FA00:  D8             MNEMR    DFB $D8
FA01:  62                      DFB $62
FA02:  5A                      DFB $5A
FA03:  48                      DFB $48
FA04:  26                      DFB $26
FA05:  62                      DFB $62
FA06:  94                      DFB $94
FA07:  88                      DFB $88
FA08:  54                      DFB $54
FA09:  44                      DFB $44
FA0A:  C8                      DFB $C8
FA0B:  54                      DFB $54
FA0C:  68                      DFB $68
FA0D:  44                      DFB $44
FA0E:  E8                      DFB $E8
FA0F:  94                      DFB $94
FA10:  00                      DFB $00
FA11:  B4                      DFB $B4
FA12:  08                      DFB $08
FA13:  84                      DFB $84
FA14:  74                      DFB $74
FA15:  B4                      DFB $B4
FA16:  28                      DFB $28
FA17:  6E                      DFB $6E
FA18:  74                      DFB $74
FA19:  F4                      DFB $F4
FA1A:  CC                      DFB $CC
FA1B:  4A                      DFB $4A
FA1C:  72                      DFB $72
FA1D:  F2                      DFB $F2
FA1E:  A4                      DFB $A4
FA1F:  8A                      DFB $8A
FA20:  00                      DFB $00
```

```
FA21: AA              DFB $AA
FA22: A2              DFB $A2
FA23: A2              DFB $A2
FA24: 74              DFB $74
FA25: 74              DFB $74
FA26: 74              DFB $74
FA27: 72              DFB $72
FA28: 44              DFB $44
FA29: 68              DFB $68
FA2A: B2              DFB $B2
FA2B: 32              DFB $32
FA2C: B2              DFB $B2
FA2D: 00              DFB $00
FA2E: 22              DFB $22
FA2F: 00              DFB $00
FA30: 1A              DFB $1A
FA31: 1A              DFB $1A
FA32: 26              DFB $26
FA33: 26              DFB $26
FA34: 72              DFB $72
FA35: 72              DFB $72
FA36: 88              DFB $88
FA37: C8              DFB $C8
FA38: C4              DFB $C4
FA39: CA              DFB $CA
FA3A: 26              DFB $26
FA3B: 48              DFB $48
FA3C: 44              DFB $44
FA3D: 44              DFB $44
FA3E: A2              DFB $A2
FA3F: C8              DFB $C8
FA40:                 PAGE
FA40: 85 45     IRQ     STA ACC
FA42: 68                PLA
FA43: 48                PHA
FA44: 0A                ASL A
FA45: 0A                ASL A
FA46: 0A                ASL A
FA47: 30 03             BMI BREAK
FA49: 6C FE 03          JMP (IRQLOC)
FA4C: 28        BREAK   PLP
FA4D: 20 4C FF          JSR SAV1
FA50: 68                PLA
FA51: 85 3A             STA PCL
FA53: 68                PLA
FA54: 85 3B             STA PCH
FA56: 6C F0 03          JMP (BRKV) ;BRKV WRITTEN OVER BY DISK BOOT
FA59: 20 82 F8  OLDBRK  JSR INSDS1
FA5C: 20 DA FA          JSR RGDSP1
FA5F: 4C 65 FF          JMP MON
FA62: D8        RESET   CLD           ; DO THIS FIRST THIS TIME
FA63: 20 84 FE          JSR SETNORM
FA66: 20 2F FB          JSR INIT
FA69: 20 93 FE          JSR SETVID
FA6C: 20 89 FE          JSR SETKBD
FA6F: AD 58 C0  INITAN  LDA SETAN0 ; AN0 = TTL HI
FA72: AD 5A C0          LDA SETAN1 ; AN1 = TTL HI
FA75: AD 5D C0          LDA CLRAN2 ; AN2 = TTL LO
FA78: AD 5F C0          LDA CLRAN3 ; AN3 = TTL LO
FA7B: AD FF CF          LDA CLRROM ; TURN OFF EXTNSN ROM
FA7E: 2C 10 C0          BIT KBDSTRB ; CLEAR KEYBOARD
FA81: D8        NEWMON  CLD
FA82: 20 3A FF          JSR BELL    ; CAUSES DELAY IF KEY BOUNCES
FA85: AD F3 03          LDA SOFTEV+1 ;IS RESET HI
FA88: 49 A5             EOR #$A5    ;A FUNNY COMPLEMENT OF THE
FA8A: CD F4 03          CMP PWREDUP ; PWR UP BYTE ???
FA8D: D0 17             BNE PWRUP   ; NO SO PWRUP
FA8F: AD F2 03          LDA SOFTEV ; YES SEE IF COLD START
FA92: D0 0F             BNE NOFIX   ; HAS BEEN DONE YET?
FA94: A9 E0             LDA #$E0    ; ??
FA96: CD F3 03          CMP SOFTEV+1 ; ??
FA99: D0 08             BNE NOFIX   ; YES SO REENTER SYSTEM
FA9B: A0 03     FIXSEV  LDY #3      ; NO SO POINT AT WARM START
```

```
FA9D  8C F2 03       STY SOFTEV ; FOR NEXT RESET
FAA0  4C 00 E0       JMP BASIC  ; AND DO THE COLD START
FAA3  6C F2 03   NOFIX    JMP (SOFTEV) ; SOFT ENTRY VECTOR
FAA6             ****************************
FAA6  20 60 FB   PWRUP    JSR APPLEII
FAA9             SETPG3   EQU *        ; SET PAGE 3 VECTORS
FAA9  A2 05               LDX #5
FAAB  BD FC FA   SETPLP   LDA PWRCON-1,X ; WITH CNTRL B ADRS
FAAE  9D EF 03            STA BRKV-1,X ; OF CURRENT BASIC
FAB1  CA                  DEX
FAB2  D0 F7               BNE SETPLP
FAB4  A9 C8               LDA #$C8     ; LOAD HI SLOT +1
FAB6  86 00               STX LOC0     ; SETPG3 MUST RETURN X=0
FAB8  85 01               STA LOC1     ; SET PTR H
FABA  A0 07      SLOOP    LDY #7       ; Y IS BYTE PTR
FABC  C6 01               DEC LOC1
FABE  A5 01               LDA LOC1
FAC0  C9 C0               CMP #$C0     ; AT LAST SLOT YET?
FAC2  F0 D7               BEQ FIXSEV ; YES AND IT CANT BE A DISK
FAC4  8D F8 07            STA MSLOT
FAC7  B1 00      NXTBYT   LDA (LOC0),Y ; FETCH A SLOT BYTE
FAC9  D9 01 FB            CMP DISKID-1,Y ; IS IT A DISK ??
FACC  D0 EC               BNE SLOOP    ; NO SO NEXT SLOT DOWN
FACE  88                  DEY
FACF  88                  DEY          ; YES SO CHECK NEXT BYTE
FAD0  10 F5               BPL NXTBYT ; UNTIL 4 CHECKED
FAD2  6C 00 00            JMP (LOC0)
FAD5  EA                  NOP
FAD6  EA                  NOP
FAD7             * REGDSP MUST ORG $FAD7
FAD7  20 8E FD   REGDSP   JSR CROUT
FADA  A9 45      RGDSP1   LDA #$45
FADC  85 40               STA A3L
FADE  A9 00               LDA #$00
FAE0  85 41               STA A3H
FAE2  A2 FB               LDX #$FB
FAE4  A9 A0      RDSP1    LDA #$A0
FAE6  20 ED FD            JSR COUT
FAE9  BD 1E FA            LDA RTBL-251,X
FAEC  20 ED FD            JSR COUT
FAEF  A9 BD               LDA #$BD
FAF1  20 ED FD            JSR COUT
FAF4             * LDA ACC+5,X
FAF4  B5 4A               DFB $B5,$4A
FAF6  20 DA FD            JSR PRBYTE
FAF9  E8                  INX
FAFA  30 E8               BMI RDSP1
FAFC  60                  RTS
FAFD  59 FA      PWRCON   DW  OLDBRK
FAFF  00 E0 45            DFB $00,$E0,$45
FB02  20 FF 00
FB05  FF         DISKID   DFB $20,$FF,$00,$FF
FB06  03 FF 3C            DFB $03,$FF,$3C
FB09  C1 D0 D0   TITLE    DFB $C1,$D0,$D0
FB0C  CC C5 A0            DFB $CC,$C5,$A0
FB0F  DD DB               DFB $DD,$DB
FB11             XLTBL    EQU *
FB11  C4 C2 C1            DFB $C4,$C2,$C1
FB14  FF C3               DFB $FF,$C3
FB16  FF FF FF            DFB $FF,$FF,$FF
FB19             *  MUST ORG $FB19
FB19  C1 D8 D9   RTBL     DFB $C1,$D8,$D9
FB1C  D0 D3               DFB $D0,$D3
FB1E  AD 70 C0   PREAD    LDA PTRIG
FB21                      LST ON
FB21  A0 00               LDY #$00
FB23  EA                  NOP
FB24  EA                  NOP
FB25  BD 64 C0   PREAD2   LDA PADDLO,X
FB28  10 04               BPL RTS2D
FB2A  C8                  INY
FB2B  D0 F8               BNE PREAD2
FB2D  88                  DEY
```

```
FB2E: 60                RTS2D    RTS
FB2F: A9 00             INIT     LDA #$00
FB31: 85 48                      STA STATUS
FB33: AD 56 C0                   LDA LORES
FB36: AD 54 C0                   LDA LOWSCR
FB39: AD 51 C0          SETTXT   LDA TXTSET
FB3C: A9 00                      LDA #$00
FB3E: F0 0B                      BEQ SETWND
FB40: AD 50 C0          SETGR    LDA TXTCLR
FB43: AD 53 C0                   LDA MIXSET
FB46: 20 36 F8                   JSR CLRTOP
FB49: A9 14                      LDA #$14
FB4B: 85 22             SETWND   STA WNDTOP
FB4D: A9 00                      LDA #$00
FB4F: 85 20                      STA WNDLFT
FB51: A9 28                      LDA #$28
FB53: 85 21                      STA WNDWDTH
FB55: A9 18                      LDA #$18
FB57: 85 23                      STA WNDBTM
FB59: A9 17                      LDA #$17
FB5B: 85 25             TABV     STA CV
FB5D: 4C 22 FC                   JMP VTAB
FB60: 20 58 FC          APPLEII  JSR HOME     ; CLEAR THE SCRN
FB63: A0 08                      LDY #8
FB65: B9 08 FB          STITLE   LDA TITLE-1,Y ; GET A CHAR
FB68: 99 0E 04                   STA LINE1+14,Y
FB6B: 88                         DEY
FB6C: D0 F7                      BNE STITLE
FB6E: 60                         RTS
FB6F: AD F3 03          SETPWRC  LDA SOFTEV+1
FB72: 49 A5                      EOR #$A5
FB74: 8D F4 03                   STA PWREDUP
FB77: 60                         RTS
FB78:                   VIDWAIT  EQU *        ; CHECK FOR A PAUSE
FB78: C9 8D                      CMP #$8D     ; ONLY WHEN I HAVE A CR
FB7A: D0 18                      BNE NOWAIT ; NOT SO, DO REGULAR
FB7C: AC 00 C0                   LDY KBD      ; IS KEY PRESSED?
FB7F: 10 13                      BPL NOWAIT ; NO
FB81: C0 93                      CPY #$93     ; IS IT CTL S ?
FB83: D0 0F                      BNE NOWAIT ; NO SO IGNORE
FB85: 2C 10 C0                   BIT KBDSTRB ; CLEAR STROBE
FB88: AC 00 C0          KBDWAIT  LDY KBD      ; WAIT TILL NEXT KEY TO RESUME
FB8B: 10 FB                      BPL KBDWAIT ; WAIT FOR KEYPRESS
FB8D: C0 83                      CPY #$83     ; IS IT CONTROL C ?
FB8F: F0 03                      BEQ NOWAIT ; YES SO LEAVE IT
FB91: 2C 10 C0                   BIT KBDSTRB ; CLR STROBE
FB94: 4C FD FB          NOWAIT   JMP VIDOUT ; DO AS BEFORE
FB97:                            PAGE
FB97: 38                ESCOLD   SEC          ; INSURE CARRY SET
FB98: 4C 2C FC                   JMP ESC1
FB9B: A8                ESCNOW   TAY          ; USE CHAR AS INDEX
FB9C: B9 48 FA                   LDA XLTBL-$C9,Y ; XLATE IJKM TO CBAD
FB9F: 20 97 FB                   JSR ESCOLD ; DO THIS CURSOR MOTION
FBA2: 20 0C FD                   JSR RDKEY    ; AND GET NEXT
FBA5: C9 CE             ESCNEW   CMP #$CE     ;IS THIS AN N ?
FBA7: B0 EE                      BCS ESCOLD ; N OR GREATER DO IT
FBA9: C9 C9                      CMP #$C9     ; LESS THAN I ?
FBAB: 90 EA                      BCC ESCOLD ; YES SO OLD WAY
FBAD: C9 CC                      CMP #$CC     ; IS IT A L ?
FBAF: F0 E6                      BEQ ESCOLD ; DO NORMAL
FBB1: D0 E8                      BNE ESCNOW ; GO DO IT
FBB3: EA                         NOP
FBB4: EA                         NOP
FBB5: EA                         NOP
FBB6: EA                         NOP
FBB7: EA                         NOP
FBB8: EA                         NOP
FBB9: EA                         NOP
FBBA: EA                         NOP
```

```
FBBB:  EA                          NOP
FBBC:  EA                          NOP
FBBD:  EA                          NOP
FBBE:  EA                          NOP
FBBF:  EA                          NOP
FBC0:  EA                          NOP
FBC1:                  *     MUST ORG $FBC1
FBC1:  48              BASCALC     PHA
FBC2:  4A                          LSR A
FBC3:  29 03                       AND #$03
FBC5:  09 04                       ORA #$04
FBC7:  85 29                       STA BASH
FBC9:  68                          PLA
FBCA:  29 18                       AND #$18
FBCC:  90 02                       BCC BASCLC2
FBCE:  69 7F                       ADC #$7F
FBD0:  85 28           BASCLC2     STA BASL
FBD2:  0A                          ASL A
FBD3:  0A                          ASL A
FBD4:  05 28                       ORA BASL
FBD6:  85 28                       STA BASL
FBD8:  60                          RTS
FBD9:  C9 87           BELL1       CMP #$87
FBDB:  D0 12                       BNE RTS2B
FBDD:  A9 40                       LDA #$40
FBDF:  20 A8 FC                    JSR WAIT
FBE2:  A0 C0                       LDY #$C0
FBE4:  A9 0C           BELL2       LDA #$0C
FBE6:  20 A8 FC                    JSR WAIT
FBE9:  AD 30 C0                    LDA SPKR
FBEC:  88                          DEY
FBED:  D0 F5                       BNE BELL2
FBEF:  60              RTS2B       RTS
FBF0:                              PAGE
FBF0:  A4 24           STORADV     LDY CH
FBF2:  91 28                       STA (BASL),Y
FBF4:  E6 24           ADVANCE     INC CH
FBF6:  A5 24                       LDA CH
FBF8:  C5 21                       CMP WNDWDTH
FBFA:  B0 66                       BCS CR
FBFC:  60              RTS3        RTS
FBFD:  C9 A0           VIDOUT      CMP #$A0
FBFF:  B0 EF                       BCS STORADV
FC01:  A8                          TAY
FC02:  10 EC                       BPL STORADV
FC04:  C9 8D                       CMP #$8D
FC06:  F0 5A                       BEQ CR
FC08:  C9 8A                       CMP #$8A
FC0A:  F0 5A                       BEQ LF
FC0C:  C9 88                       CMP #$88
FC0E:  D0 C9                       BNE BELL1
FC10:  C6 24           BS          DEC CH
FC12:  10 E8                       BPL RTS3
FC14:  A5 21                       LDA WNDWDTH
FC16:  85 24                       STA CH
FC18:  C6 24                       DEC CH
FC1A:  A5 22           UP          LDA WNDTOP
FC1C:  C5 25                       CMP CV
FC1E:  B0 0B                       BCS RTS4
FC20:  C6 25                       DEC CV
FC22:  A5 25           VTAB        LDA CV
FC24:  20 C1 FB        VTABZ       JSR BASCALC
FC27:  65 20                       ADC WNDLFT
FC29:  85 28                       STA BASL
FC2B:  60              RTS4        RTS
FC2C:  49 C0           ESC1        EOR #$C0     ; ESC @ ?
FC2E:  F0 28                       BEQ HOME     ; IF SO DO HOME AND CLEAR
FC30:  69 FD                       ADC #$FD     ; ESC-A OR B CHECK
FC32:  90 C0                       BCC ADVANCE  ; A, ADVANCE
FC34:  F0 DA                       BEQ BS       ; B, BACKSPACE
FC36:  69 FD                       ADC #$FD     ; ESC-C OR D CHECK
FC38:  90 2C                       BCC LF       ; C, DOWN
FC3A:  F0 DE                       BEQ UP       ; D, GO UP
```

```
FC3C:  69 FD                   ADC #$FD     ; ESC-E OR F CKECK
FC3E:  90 5C                   BCC CLREOL   ; E, CLEAR TO END OF LINE
FC40:  D0 E9                   BNE RTS4     ; ELSE NOT F,RETURN
FC42:  A4 24        CLREOP     LDY CH       ; ESC F IS CLR TO END OF PAGE
FC44:  A5 25                   LDA CV
FC46:  48           CLEOP1     PHA
FC47:  20 24 FC                JSR VTABZ
FC4A:  20 9E FC                JSR CLEOLZ
FC4D:  A0 00                   LDY #$00
FC4F:  68                      PLA
FC50:  69 00                   ADC #$00
FC52:  C5 23                   CMP WNDBTM
FC54:  90 F0                   BCC CLEOP1
FC56:  B0 CA                   BCS VTAB
FC58:  A5 22        HOME       LDA WNDTOP
FC5A:  85 25                   STA CV
FC5C:  A0 00                   LDY #$00
FC5E:  84 24                   STY CH
FC60:  F0 E4                   BEQ CLEOP1
FC62:                          PAGE
FC62:  A9 00        CR         LDA #$00
FC64:  85 24                   STA CH
FC66:  E6 25        LF         INC CV
FC68:  A5 25                   LDA CV
FC6A:  C5 23                   CMP WNDBTM
FC6C:  90 B6                   BCC VTABZ
FC6E:  C6 25                   DEC CV
FC70:  A5 22        SCROLL     LDA WNDTOP
FC72:  48                      PHA
FC73:  20 24 FC                JSR VTABZ
FC76:  A5 28        SCRL1      LDA BASL
FC78:  85 2A                   STA BAS2L
FC7A:  A5 29                   LDA BASH
FC7C:  85 2B                   STA BAS2H
FC7E:  A4 21                   LDY WNDWDTH
FC80:  88                      DEY
FC81:  68                      PLA
FC82:  69 01                   ADC #$01
FC84:  C5 23                   CMP WNDBTM
FC86:  B0 0D                   BCS SCRL3
FC88:  48                      PHA
FC89:  20 24 FC                JSR VTABZ
FC8C:  B1 28        SCRL2      LDA (BASL),Y
FC8E:  91 2A                   STA (BAS2L),Y
FC90:  88                      DEY
FC91:  10 F9                   BPL SCRL2
FC93:  30 E1                   BMI SCRL1
FC95:  A0 00        SCRL3      LDY #$00
FC97:  20 9E FC                JSR CLEOLZ
FC9A:  B0 86                   BCS VTAB
FC9C:  A4 24        CLREOL     LDY CH
FC9E:  A9 A0        CLEOLZ     LDA #$A0
FCA0:  91 28        CLEOL2     STA (BASL),Y
FCA2:  C8                      INY
FCA3:  C4 21                   CPY WNDWDTH
FCA5:  90 F9                   BCC CLEOL2
FCA7:  60                      RTS
FCA8:  38           WAIT       SEC
FCA9:  48           WAIT2      PHA
FCAA:  E9 01        WAIT3      SBC #$01
FCAC:  D0 FC                   BNE WAIT3
FCAE:  68                      PLA
FCAF:  E9 01                   SBC #$01
FCB1:  D0 F6                   BNE WAIT2
FCB3:  60                      RTS
FCB4:  E6 42        NXTA4      INC A4L
FCB6:  D0 02                   BNE NXTA1
FCB8:  E6 43                   INC A4H
FCBA:  A5 3C        NXTA1      LDA A1L
FCBC:  C5 3E                   CMP A2L
FCBE:  A5 3D                   LDA A1H
FCC0:  E5 3F                   SBC A2H
FCC2:  E6 3C                   INC A1L
```

```
FCC4:  D0 02              BNE RTS4B
FCC6:  E6 3D              INC A1H
FCC8:  60        RTS4B    RTS
FCC9:                     PAGE
FCC9:  A0 4B     HEADR    LDY #$4B
FCCB:  20 DB FC           JSR ZERDLY
FCCE:  D0 F9              BNE HEADR
FCD0:  69 FE              ADC #$FE
FCD2:  B0 F5              BCS HEADR
FCD4:  A0 21              LDY #$21
FCD6:  20 DB FC  WRBIT    JSR ZERDLY
FCD9:  C8                 INY
FCDA:  C8                 INY
FCDB:  88        ZERDLY   DEY
FCDC:  D0 FD              BNE ZERDLY
FCDE:  90 05              BCC WRTAPE
FCE0:  A0 32              LDY #$32
FCE2:  88        ONEDLY   DEY
FCE3:  D0 FD              BNE ONEDLY
FCE5:  AC 20 C0  WRTAPE   LDY TAPEOUT
FCE8:  A0 2C              LDY #$2C
FCEA:  CA                 DEX
FCEB:  60                 RTS
FCEC:  A2 08     RDBYTE   LDX #$08
FCEE:  48        RDBYT2   PHA
FCEF:  20 FA FC           JSR RD2BIT
FCF2:  68                 PLA
FCF3:  2A                 ROL A
FCF4:  A0 3A              LDY #$3A
FCF6:  CA                 DEX
FCF7:  D0 F5              BNE RDBYT2
FCF9:  60                 RTS
FCFA:  20 FD FC  RD2BIT   JSR RDBIT
FCFD:  88        RDBIT    DEY
FCFE:  AD 60 C0           LDA TAPEIN
FD01:  45 2F              EOR LASTIN
FD03:  10 F8              BPL RDBIT
FD05:  45 2F              EOR LASTIN
FD07:  85 2F              STA LASTIN
FD09:  C0 80              CPY #$80
FD0B:  60                 RTS
FD0C:  A4 24     RDKEY    LDY CH
FD0E:  B1 28              LDA (BASL),Y
FD10:  48                 PHA
FD11:  29 3F              AND #$3F
FD13:  09 40              ORA #$40
FD15:  91 28              STA (BASL),Y
FD17:  68                 PLA
FD18:  6C 38 00           JMP (KSWL)
FD1B:  E6 4E     KEYIN    INC RNDL
FD1D:  D0 02              BNE KEYIN2
FD1F:  E6 4F              INC RNDH
FD21:  2C 00 C0  KEYIN2   BIT KBD       ; READ KEYBOARD
FD24:  10 F5              BPL KEYIN
FD26:  91 28              STA (BASL),Y
FD28:  AD 00 C0           LDA KBD
FD2B:  2C 10 C0           BIT KBDSTRB
FD2E:  60                 RTS
FD2F:  20 0C FD  ESC      JSR RDKEY
FD32:  20 A5 FB           JSR ESCNEW
FD35:  20 0C FD  RDCHAR   JSR RDKEY
FD38:  C9 9B              CMP #$9B
FD3A:  F0 F3              BEQ ESC
FD3C:  60                 RTS
FD3D:                     PAGE
FD3D:  A5 32     NOTCR    LDA INVFLG
FD3F:  48                 PHA
FD40:  A9 FF              LDA #$FF
FD42:  85 32              STA INVFLG
FD44:  BD 00 02           LDA IN,X
FD47:  20 ED FD           JSR COUT
FD4A:  68                 PLA
FD4B:  85 32              STA INVFLG
```

```
FD4D: BD 00 02               LDA IN,X
FD50: C9 88                  CMP #$88
FD52: F0 1D                  BEQ BCKSPC
FD54: C9 98                  CMP #$98
FD56: F0 0A                  BEQ CANCEL
FD58: E0 F8                  CPX #$F8
FD5A: 90 03                  BCC NOTCR1
FD5C: 20 3A FF               JSR BELL
FD5F: E8           NOTCR1    INX
FD60: D0 13                  BNE NXTCHAR
FD62: A9 DC        CANCEL    LDA #$DC
FD64: 20 ED FD               JSR COUT
FD67: 20 8E FD     GETLNZ    JSR CROUT
FD6A: A5 33        GETLN     LDA PROMPT
FD6C: 20 ED FD               JSR COUT
FD6F: A2 01                  LDX #$01
FD71: 8A           BCKSPC    TXA
FD72: F0 F3                  BEQ GETLNZ
FD74: CA                     DEX
FD75: 20 35 FD     NXTCHAR   JSR RDCHAR
FD78: C9 95                  CMP #$95
FD7A: D0 02                  BNE CAPTST
FD7C: B1 28                  LDA (BASL),Y
FD7E: C9 E0        CAPTST    CMP #$E0
FD80: 90 02                  BCC ADDINP
FD82: 29 DF                  AND #$DF      ; SHIFT TO UPPER CASE
FD84: 9D 00 02     ADDINP    STA IN,X
FD87: C9 8D                  CMP #$8D
FD89: D0 B2                  BNE NOTCR
FD8B: 20 9C FC               JSR CLREOL
FD8E: A9 8D        CROUT     LDA #$8D
FD90: D0 5B                  BNE COUT
FD92: A4 3D        PRA1      LDY A1H
FD94: A6 3C                  LDX A1L
FD96: 20 8E FD     PRYX2     JSR CROUT
FD99: 20 40 F9               JSR PRNTYX
FD9C: A0 00                  LDY #$00
FD9E: A9 AD                  LDA #$AD
FDA0: 4C ED FD               JMP COUT
FDA3:                        PAGE
FDA3: A5 3C        XAM8      LDA A1L
FDA5: 09 07                  ORA #$07
FDA7: 85 3E                  STA A2L
FDA9: A5 3D                  LDA A1H
FDAB: 85 3F                  STA A2H
FDAD: A5 3C        MOD8CHK   LDA A1L
FDAF: 29 07                  AND #$07
FDB1: D0 03                  BNE DATAOUT
FDB3: 20 92 FD     XAM       JSR PRA1
FDB6: A9 A0        DATAOUT   LDA #$A0
FDB8: 20 ED FD               JSR COUT
FDBB: B1 3C                  LDA (A1L),Y
FDBD: 20 DA FD               JSR PRBYTE
FDC0: 20 BA FC               JSR NXTA1
FDC3: 90 E8                  BCC MOD8CHK
FDC5: 60           RTS4C     RTS
FDC6: 4A           XAMPM     LSR A
FDC7: 90 EA                  BCC XAM
FDC9: 4A                     LSR A
FDCA: 4A                     LSR A
FDCB: A5 3E                  LDA A2L
FDCD: 90 02                  BCC ADD
FDCF: 49 FF                  EOR #$FF
FDD1: 65 3C        ADD       ADC A1L
FDD3: 48                     PHA
FDD4: A9 BD                  LDA #$BD
FDD6: 20 ED FD               JSR COUT
FDD9: 68                     PLA
FDDA: 48           PRBYTE    PHA
FDDB: 4A                     LSR A
FDDC: 4A                     LSR A
FDDD: 4A                     LSR A
FDDE: 4A                     LSR A
```

```
FDDF:  20 E5 FD               JSR PRHEXZ
FDE2:  68                     PLA
FDE3:  29 0F      PRHEX       AND #$0F
FDE5:  09 B0      PRHEXZ      ORA #$B0
FDE7:  C9 BA                  CMP #$BA
FDE9:  90 02                  BCC COUT
FDEB:  69 06                  ADC #$06
FDED:  6C 36 00   COUT        JMP (CSWL)
FDF0:  C9 A0      COUT1       CMP #$A0
FDF2:  90 02                  BCC COUTZ
FDF4:  25 32                  AND INVFLG
FDF6:  84 35      COUTZ       STY YSAV1
FDF8:  48                     PHA
FDF9:  20 78 FB               JSR VIDWAIT ; GO CHECK FOR PAUSE
FDFC:  68                     PLA
FDFD:  A4 35                  LDY YSAV1
FDFF:  60                     RTS
FE00:                         PAGE
FE00:  C6 34      BL1         DEC YSAV
FE02:  F0 9F                  BEQ XAM8
FE04:  CA         BLANK       DEX
FE05:  D0 16                  BNE SETMDZ
FE07:  C9 BA                  CMP #$BA
FE09:  D0 BB                  BNE XAMPM
FE0B:  85 31      STOR        STA MODE
FE0D:  A5 3E                  LDA A2L
FE0F:  91 40                  STA (A3L),Y
FE11:  E6 40                  INC A3L
FE13:  D0 02                  BNE RTS5
FE15:  E6 41                  INC A3H
FE17:  60         RTS5        RTS
FE18:  A4 34      SETMODE     LDY YSAV
FE1A:  B9 FF 01               LDA IN-1,Y
FE1D:  85 31      SETMDZ      STA MODE
FE1F:  60                     RTS
FE20:  A2 01      LT          LDX #$01
FE22:  B5 3E      LT2         LDA A2L,X
FE24:  95 42                  STA A4L,X
FE26:  95 44                  STA A5L,X
FE28:  CA                     DEX
FE29:  10 F7                  BPL LT2
FE2B:  60                     RTS
FE2C:  B1 3C      MOVE        LDA (A1L),Y
FE2E:  91 42                  STA (A4L),Y
FE30:  20 B4 FC               JSR NXTA4
FE33:  90 F7                  BCC MOVE
FE35:  60                     RTS
FE36:  B1 3C      VFY         LDA (A1L),Y
FE38:  D1 42                  CMP (A4L),Y
FE3A:  F0 1C                  BEQ VFYOK
FE3C:  20 92 FD               JSR PRA1
FE3F:  B1 3C                  LDA (A1L),Y
FE41:  20 DA FD               JSR PRBYTE
FE44:  A9 A0                  LDA #$A0
FE46:  20 ED FD               JSR COUT
FE49:  A9 A8                  LDA #$A8
FE4B:  20 ED FD               JSR COUT
FE4E:  B1 42                  LDA (A4L),Y
FE50:  20 DA FD               JSR PRBYTE
FE53:  A9 A9                  LDA #$A9
FE55:  20 ED FD               JSR COUT
FE58:  20 B4 FC   VFYOK       JSR NXTA4
FE5B:  90 D9                  BCC VFY
FE5D:  60                     RTS
FE5E:  20 75 FE   LIST        JSR A1PC
FE61:  A9 14                  LDA #$14
FE63:  48         LIST2       PHA
FE64:  20 D0 F8               JSR INSTDSP
FE67:  20 53 F9               JSR PCADJ
FE6A:  85 3A                  STA PCL
FE6C:  84 3B                  STY PCH
FE6E:  68                     PLA
FE6F:  38                     SEC
```

```
FE70:  E9 01               SBC #$01
FE72:  D0 EF               BNE LIST2
FE74:  60                  RTS
FE75:                      PAGE
FE75:  8A        A1PC      TXA
FE76:  F0 07               BEQ A1PCRTS
FE78:  B5 3C     A1PCLP    LDA A1L,X
FE7A:  95 3A               STA PCL,X
FE7C:  CA                  DEX
FE7D:  10 F9               BPL A1PCLP
FE7F:  60        A1PCRTS   RTS
FE80:  A0 3F     SETINV    LDY #$3F
FE82:  D0 02               BNE SETIFLG
FE84:  A0 FF     SETNORM   LDY #$FF
FE86:  84 32     SETIFLG   STY INVFLG
FE88:  60                  RTS
FE89:  A9 00     SETKBD    LDA #$00
FE8B:  85 3E     INPORT    STA A2L
FE8D:  A2 38     INPRT     LDX #KSWL
FE8F:  A0 1B               LDY #KEYIN
FE91:  D0 08               BNE IOPRT
FE93:  A9 00     SETVID    LDA #$00
FE95:  85 3E     OUTPORT   STA A2L
FE97:  A2 36     OUTPRT    LDX #CSWL
FE99:  A0 F0               LDY #COUT1
FE9B:  A5 3E     IOPRT     LDA A2L
FE9D:  29 0F               AND #$0F
FE9F:  F0 06               BEQ IOPRT1
FEA1:  09 C0               ORA #IOADR/256
FEA3:  A0 00               LDY #$00
FEA5:  F0 02               BEQ IOPRT2
FEA7:  A9 FD     IOPRT1    LDA #COUT1/256
FEA9:            IOPRT2    EQU *
FEA9:  94 00               STY LOC0,X ; $94,$00
FEAB:  95 01               STA LOC1,X ; $95,$01
FEAD:  60                  RTS
FEAE:  EA                  NOP
FEAF:  EA                  NOP
FEB0:  4C 00 E0  XBASIC    JMP BASIC
FEB3:  4C 03 E0  BASCONT   JMP BASIC2
FEB6:  20 75 FE  GO        JSR A1PC
FEB9:  20 3F FF            JSR RESTORE
FEBC:  6C 3A 00            JMP (PCL)
FEBF:  4C D7 FA  REGZ      JMP REGDSP
FEC2:  60        TRACE     RTS
FEC3:            * TRACE IS GONE
FEC3:  EA                  NOP
FEC4:  60        STEPZ     RTS          ; STEP IS GONE
FEC5:  EA                  NOP
FEC6:  EA                  NOP
FEC7:  EA                  NOP
FEC8:  EA                  NOP
FEC9:  EA                  NOP
FECA:  4C F8 03  USR       JMP USRADR
FECD:                      PAGE
FECD:  A9 40     WRITE     LDA #$40
FECF:  20 C9 FC            JSR HEADR
FED2:  A0 27               LDY #$27
FED4:  A2 00     WR1       LDX #$00
FED6:  41 3C               EOR (A1L,X)
FED8:  48                  PHA
FED9:  A1 3C               LDA (A1L,X)
FEDB:  20 ED FE            JSR WRBYTE
FEDE:  20 BA FC            JSR NXTA1
FEE1:  A0 1D               LDY #$1D
FEE3:  68                  PLA
FEE4:  90 EE               BCC WR1
FEE6:  A0 22               LDY #$22
FEE8:  20 ED FE            JSR WRBYTE
FEEB:  F0 4D               BEQ BELL
FEED:  A2 10     WRBYTE    LDX #$10
FEEF:  0A        WRBYT2    ASL A
FEF0:  20 D6 FC            JSR WRBIT
```

```
FEF3:  D0 FA                       BNE WRBYT2
FEF5:  60                          RTS
FEF6:  20 00 FE       CRMON        JSR BL1
FEF9:  68                          PLA
FEFA:  68                          PLA
FEFB:  D0 6C                       BNE MONZ
FEFD:  20 FA FC       READ         JSR RD2BIT
FF00:  A9 16                       LDA #$16
FF02:  20 C9 FC                    JSR HEADR
FF05:  85 2E                       STA CHKSUM
FF07:  20 FA FC                    JSR RD2BIT
FF0A:  A0 24          RD2          LDY #$24
FF0C:  20 FD FC                    JSR RDBIT
FF0F:  B0 F9                       BCS RD2
FF11:  20 FD FC                    JSR RDBIT
FF14:  A0 3B                       LDY #$3B
FF16:  20 EC FC       RD3          JSR RDBYTE
FF19:  81 3C                       STA (A1L,X)
FF1B:  45 2E                       EOR CHKSUM
FF1D:  85 2E                       STA CHKSUM
FF1F:  20 BA FC                    JSR NXTA1
FF22:  A0 35                       LDY #$35
FF24:  90 F0                       BCC RD3
FF26:  20 EC FC                    JSR RDBYTE
FF29:  C5 2E                       CMP CHKSUM
FF2B:  F0 0D                       BEQ BELL
FF2D:  A9 C5          PRERR        LDA #$C5
FF2F:  20 ED FD                    JSR COUT
FF32:  A9 D2                       LDA #$D2
FF34:  20 ED FD                    JSR COUT
FF37:  20 ED FD                    JSR COUT
FF3A:  A9 87          BELL         LDA #$87
FF3C:  4C ED FD                    JMP COUT
FF3F:                              PAGE
FF3F:  A5 48          RESTORE      LDA STATUS
FF41:  48                          PHA
FF42:  A5 45                       LDA A5H
FF44:  A6 46          RESTR1       LDX XREG
FF46:  A4 47                       LDY YREG
FF48:  28                          PLP
FF49:  60                          RTS
FF4A:  85 45          SAVE         STA A5H
FF4C:  86 46          SAV1         STX XREG
FF4E:  84 47                       STY YREG
FF50:  08                          PHP
FF51:  68                          PLA
FF52:  85 48                       STA STATUS
FF54:  BA                          TSX
FF55:  86 49                       STX SPNT
FF57:  D8                          CLD
FF58:  60                          RTS
FF59:  20 84 FE       OLDRST       JSR SETNORM
FF5C:  20 2F FB                    JSR INIT
FF5F:  20 93 FE                    JSR SETVID
FF62:  20 89 FE                    JSR SETKBD
FF65:                              PAGE
FF65:  D8             MON          CLD
FF66:  20 3A FF                    JSR BELL
FF69:  A9 AA          MONZ         LDA #$AA
FF6B:  85 33                       STA PROMPT
FF6D:  20 67 FD                    JSR GETLNZ
FF70:  20 C7 FF                    JSR ZMODE
FF73:  20 A7 FF       NXTITM       JSR GETNUM
FF76:  84 34                       STY YSAV
FF78:  A0 17                       LDY #$17
FF7A:  88             CHRSRCH      DEY
FF7B:  30 E8                       BMI MON
FF7D:  D9 CC FF                    CMP CHRTBL,Y
FF80:  D0 F8                       BNE CHRSRCH
FF82:  20 BE FF                    JSR TOSUB
FF85:  A4 34                       LDY YSAV
FF87:  4C 73 FF                    JMP NXTITM
FF8A:  A2 03          DIG          LDX #$03
```

```
FF8C:  0A                        ASL A
FF8D:  0A                        ASL A
FF8E:  0A                        ASL A
FF8F:  0A                        ASL A
FF90:  0A              NXTBIT    ASL A
FF91:  26 3E                     ROL A2L
FF93:  26 3F                     ROL A2H
FF95:  CA                        DEX
FF96:  10 F8                     BPL NXTBIT
FF98:  A5 31           NXTBAS    LDA MODE
FF9A:  D0 06                     BNE NXTBS2
FF9C:                  *
FF9C:  B5 3F                     LDA A2H, X
FF9E:                  *
FF9E:  95 3D                     STA A1H, X
FFA0:                  *
FFA0:  95 41                     STA A3H, X
FFA2:  E8              NXTBS2    INX
FFA3:  F0 F3                     BEQ NXTBAS
FFA5:  D0 06                     BNE NXTCHR
FFA7:  A2 00           GETNUM    LDX #$00
FFA9:  86 3E                     STX A2L
FFAB:  86 3F                     STX A2H
FFAD:  B9 00 02        NXTCHR    LDA IN, Y
FFB0:  C8                        INY
FFB1:  49 B0                     EOR #$B0
FFB3:  C9 0A                     CMP #$0A
FFB5:  90 D3                     BCC DIG
FFB7:  69 88                     ADC #$88
FFB9:  C9 FA                     CMP #$FA
FFBB:  B0 CD                     BCS DIG
FFBD:  60                        RTS
FFBE:  A9 FE           TOSUB     LDA #GO/256
FFC0:  48                        PHA
FFC1:  B9 E3 FF                  LDA SUBTBL, Y
FFC4:  48                        PHA
FFC5:  A5 31                     LDA MODE
FFC7:  A0 00           ZMODE     LDY #$00
FFC9:  84 31                     STY MODE
FFCB:  60                        RTS
FFCC:                            PAGE
FFCC:  BC              CHRTBL    DFB $BC
FFCD:  B2                        DFB $B2
FFCE:  BE                        DFB $BE
FFCF:  B2                        DFB $B2       ; T CMD NOW LIKE USR
FFD0:  EF                        DFB $EF
FFD1:  C4                        DFB $C4
FFD2:  B2                        DFB $B2       ; S CMD NOW LIKE USR
FFD3:  A9                        DFB $A9
FFD4:  BB                        DFB $BB
FFD5:  A6                        DFB $A6
FFD6:  A4                        DFB $A4
FFD7:  06                        DFB $06
FFD8:  95                        DFB $95
FFD9:  07                        DFB $07
FFDA:  02                        DFB $02
FFDB:  05                        DFB $05
FFDC:  F0                        DFB $F0
FFDD:  00                        DFB $00
FFDE:  EB                        DFB $EB
FFDF:  93                        DFB $93
FFE0:  A7                        DFB $A7
FFE1:  C6                        DFB $C6
FFE2:  99                        DFB $99
FFE3:  B2              SUBTBL    DFB $B2
FFE4:  C9                        DFB $C9
FFE5:  BE                        DFB $BE
FFE6:  C1                        DFB $C1
FFE7:  35                        DFB $35
FFE8:  8C                        DFB $8C
FFE9:  C4                        DFB $C4
FFEA:  96                        DFB $96
FFEB:  AF                        DFB $AF
```

```
FFEC:  17          DFB $17
FFED:  17          DFB $17
FFEE:  2B          DFB $2B
FFEF:  1F          DFB $1F
FFF0:  83          DFB $83
FFF1:  7F          DFB $7F
FFF2:  5D          DFB $5D
FFF3:  CC          DFB $CC
FFF4:  B5          DFB $B5
FFF5:  FC          DFB $FC
FFF6:  17          DFB $17
FFF7:  17          DFB $17
FFF8:  F5          DFB $F5
FFF9:  03          DFB $03
FFFA:  FB 03       DW  NMI
FFFC:  62 FA       DW  RESET
FFFE:  40 FA       DW  IRQ

ENDASM
```

MONITOR ROMin LISTAUS

```
***************************
*                         *
*        APPLE II         *
*      SYSTEM MONITOR     *
*                         *
*    COPYRIGHT 1977 BY    *
*   APPLE COMPUTER, INC.  *
*                         *
*   ALL RIGHTS RESERVED   *
*                         *
*       S. WOZNIAK        *
*        A. BAUM          *
*                         *
***************************
          TITLE              "APPLE II SYSTEM MONITOR"
LOC0       EPZ   $00
LOC1       EPZ   $01
WNDLFT     EPZ   $20
WNDWDTH    EPZ   $21
WNDTOP     EPZ   $22
WNDBTM     EPZ   $23
CH         EPZ   $24
CV         EPZ   $25
GBASL      EPZ   $26
GBASH      EPZ   $27
BASL       EPZ   $28
BASH       EPZ   $29
BAS2L      EPZ   $2A
BAS2H      EPZ   $2B
H2         EPZ   $2C
LMNEM      EPZ   $2C
RTNL       EPZ   $2C
V2         EPZ   $2D
RMNEM      EPZ   $2D
RTNH       EPZ   $2D
MASK       EPZ   $2E
CHKSUM     EPZ   $2E
FORMAT     EPZ   $2E
LASTIN     EPZ   $2F
LENGTH     EPZ   $2F
SIGN       EPZ   $2F
COLOR      EPZ   $30
MODE       EPZ   $31
INVFLG     EPZ   $32
PROMPT     EPZ   $33
YSAV       EPZ   $34
YSAV1      EPZ   $35
CSWL       EPZ   $36
CSWH       EPZ   $37
KSWL       EPZ   $38
KSWH       EPZ   $39
PCL        EPZ   $3A
PCH        EPZ   $3B
XQT        EPZ   $3C
A1L        EPZ   $3C
A1H        EPZ   $3D
A2L        EPZ   $3E
A2H        EPZ   $3F
A3L        EPZ   $40
A3H        EPZ   $41
A4L        EPZ   $42
A4H        EPZ   $43
A5L        EPZ   $44
A5H        EPZ   $45
ACC        EPZ   $45
XREG       EPZ   $46
YREG       EPZ   $47
STATUS     EPZ   $48
```

```
                      SPNT      EPZ  $49
                      RNDL      EPZ  $4E
                      RNDH      EPZ  $4F
                      ACL       EPZ  $50
                      ACH       EPZ  $51
                      XTNDL     EPZ  $52
                      XTNDH     EPZ  $53
                      AUXL      EPZ  $54
                      AUXH      EPZ  $55
                      PICK      EPZ  $95
                      IN        EQU  $0200
                      USRADR    EQU  $03F8
                      NMI       EQU  $03FB
                      IRQLOC    EQU  $03FE
                      IOADR     EQU  $C000
                      KBD       EQU  $C000
                      KBDSTRB   EQU  $C010
                      TAPEOUT   EQU  $C020
                      SPKR      EQU  $C030
                      TXTCLR    EQU  $C050
                      TXTSET    EQU  $C051
                      MIXCLR    EQU  $C052
                      MIXSET    EQU  $C053
                      LOWSCR    EQU  $C054
                      HISCR     EQU  $C055
                      LORES     EQU  $C056
                      HIRES     EQU  $C057
                      TAPEIN    EQU  $C060
                      PADDL0    EQU  $C064
                      PTRIG     EQU  $C070
                      BASIC     EQU  $E000
                      BASIC2    EQU  $E003
                                ORG  $F800           ROM START ADDRESS
F800:  4A             PLOT      LSR  A               Y-COORD/2
F801:  08                       PHP                  SAVE LSB IN CARRY
F802:  20 47 F8                 JSR  GBASCALC        CALC BASE ADR IN GBASL,H
F805:  28                       PLP                  RESTORE LSB FROM CARRY
F806:  A9 0F                    LDA  #$0F            MASK $0F IF EVEN
F808:  90 02                    BCC  RTMASK
F80A:  69 E0                    ADC  #$E0            MASK $F0 IF ODD
F80C:  85 2E          RTMASK    STA  MASK
F80E:  B1 26          PLOT1     LDA  (GBASL),Y       DATA
F810:  45 30                    EOR  COLOR            XOR  COLOR
F812:  25 2E                    AND  MASK              AND MASK
F814:  51 26                    EOR  (GBASL),Y          XOR DATA
F816:  91 26                    STA  (GBASL),Y           TO DATA
F818:  60                       RTS
F819:  20 00 F8       HLINE     JSR  PLOT            PLOT SQUARE
F81C:  C4 2C          HLINE1    CPY  H2              DONE?
F81E:  B0 11                    BCS  RTS1             YES, RETURN
F820:  C8                       INY                   NO, INCR INDEX (X-COORD)
F821:  20 0E F8                 JSR  PLOT1           PLOT NEXT SQUARE
F824:  90 F6                    BCC  HLINE1          ALWAYS TAKEN
F826:  69 01          VLINEZ    ADC  #$01            NEXT Y-COORD
F828:  48             VLINE     PHA                   SAVE ON STACK
F829:  20 00 F8                 JSR  PLOT             PLOT SQUARE
F82C:  68                       PLA
F82D:  C5 2D                    CMP  V2              DONE?
F82F:  90 F5                    BCC  VLINEZ           NO,LOOP.
F831:  60             RTS1      RTS
F832:  A0 2F          CLRSCR    LDY  #$2F            MAX Y, FULL SCRN CLR
F834:  D0 02                    BNE  CLRSC2          ALWAYS TAKEN
F836:  A0 27          CLRTOP    LDY  #$27            MAX Y, TOP SCRN CLR
F838:  84 2D          CLRSC2    STY  V2              STORE AS BOTTOM COORD
                      *                          FOR VLINE CALLS
F83A:  A0 27                    LDY  #$27            RIGHTMOST X-COORD (COLUMN)
F83C:  A9 00          CLRSC3    LDA  #$0             TOP COORD FOR VLINE CALLS
F83E:  85 30                    STA  COLOR           CLEAR COLOR (BLACK)
F840:  20 28 F8                 JSR  VLINE           DRAW VLINE
F843:  88                       DEY                  NEXT LEFTMOST X-COORD
F844:  10 F6                    BPL  CLRSC3          LOOP UNTIL DONE.
F846:  60                       RTS
F847:  48             GBASCALC  PHA                  FOR INPUT 000DEFGH
F848:  4A                       LSR  A
```

```
F849:  29 03               AND  #$03
F84B:  09 04               ORA  #$04         GENERATE GBASH=000001FG
F84D:  85 27               STA  GBASH
F84F:  68                  PLA                AND GBASL=HDEDE000
F850:  29 18               AND  #$18
F852:  90 02               BCC  GBCALC
F854:  69 7F               ADC  #$7F
F856:  85 26     GBCALC    STA  GBASL
F858:  0A                  ASL  A
F859:  0A                  ASL  A
F85A:  05 26               ORA  GBASL
F85C:  85 26               STA  GBASL
F85E:  60                  RTS
F85F:  A5 30     NXTCOL    LDA  COLOR        INCREMENT COLOR BY 3
F861:  18                  CLC
F862:  69 03               ADC  #$03
F864:  29 0F     SETCOL    AND  #$0F         SETS COLOR=17*A MOD 16
F866:  85 30               STA  COLOR
F868:  0A                  ASL  A            BOTH HALF BYTES OF COLOR EQUAL
F869:  0A                  ASL  A
F86A:  0A                  ASL  A
F86B:  0A                  ASL  A
F86C:  05 30               ORA  COLOR
F86E:  85 30               STA  COLOR
F870:  60                  RTS
F871:  4A        SCRN      LSR  A            READ SCREEN Y-COORD/2
F872:  08                  PHP               SAVE LSB (CARRY)
F873:  20 47 F8            JSR  GBASCALC     CALC BASE ADDRESS
F876:  B1 26               LDA  (GBASL),Y    GET BYTE
F878:  28                  PLP               RESTORE LSB FROM CARRY
F879:  90 04     SCRN2     BCC  RTMSKZ       IF EVEN, USE LO H
F87B:  4A                  LSR  A
F87C:  4A                  LSR  A
F87D:  4A                  LSR  A            SHIFT HIGH HALF BYTE DOWN
F87E:  4A                  LSR  A
F87F:  29 0F     RTMSKZ    AND  #$0F         MASK 4-BITS
F881:  60                  RTS
F882:  A6 3A     INSDS1    LDX  PCL          PRINT PCL,H
F884:  A4 3B               LDY  PCH
F886:  20 96 FD            JSR  PRYX2
F889:  20 48 F9            JSR  PRBLNK       FOLLOWED BY A BLANK
F88C:  A1 3A               LDA  (PCL,X)      GET OP CODE
F88E:  A8        INSDS2    TAY
F88F:  4A                  LSR  A            EVEN/ODD TEST
F890:  90 09               BCC  IEVEN
F892:  6A                  ROR  A            BIT 1 TEST
F893:  B0 10               BCS  ERR          XXXXXX11 INVALID OP
F895:  C9 A2               CMP  #$A2
F897:  F0 0C               BEQ  ERR          OPCODE $89 INVALID
F899:  29 87               AND  #$87         MASK BITS
F89B:  4A        IEVEN     LSR  A            LSB INTO CARRY FOR L/R TEST
F89C:  AA                  TAX
F89D:  BD 62 F9            LDA  FMT1,X       GET FORMAT INDEX BYTE
F8A0:  20 79 F8            JSR  SCRN2        R/L H-BYTE ON CARRY
F8A3:  D0 04               BNE  GETFMT
F8A5:  A0 80     ERR       LDY  #$80         SUBSTITUTE $80 FOR INVALID OPS
F8A7:  A9 00               LDA  #$0          SET PRINT FORMAT INDEX TO 0
F8A9:  AA        GETFMT    TAX
F8AA:  BD A6 F9            LDA  FMT2,X       INDEX INTO PRINT FORMAT TABLE
F8AD:  85 2E               STA  FORMAT       SAVE FOR ADR FIELD FORMATTING
F8AF:  29 03               AND  #$03         MASK FOR 2-BIT LENGTH
                 *         (P=1 BYTE, 1=2 BYTE, 2=3 BYTE)
F8B1:  85 2F               STA  LENGTH
F8B3:  98                  TYA               OPCODE
F8B4:  29 8F               AND  #$8F         MASK FOR 1XXX1010 TEST
F8B6:  AA                  TAX                SAVE IT
F8B7:  98                  TYA               OPCODE TO A AGAIN
F8B8:  A0 03               LDY  #$03
F8BA:  E0 8A               CPX  #$8A
F8BC:  F0 0B               BEQ  MNNDX3
F8BE:  4A        MNNDX1    LSR  A
F8BF:  90 08               BCC  MNNDX3       FORM INDEX INTO MNEMONIC TABLE
F8C1:  4A                  LSR  A
```

```
F8C2:  4A        MNNDX2   LSR  A               1) 1XXX1010=>00101XXX
F8C3:  09 20              ORA  #$20            2) XXXYYY01=>00111XXX
F8C5:  88                 DEY                   3) XXXYYY10=>00110XXX
F8C6:  D0 FA              BNE  MNNDX2          4) XXXYY100=>00100XXX
F8C8:  C8                 INY                   5) XXXXX000=>000XXXXX
F8C9:  88        MNNDX3   DEY
F8CA:  D0 F2              BNE  MNNDX1
F8CC:  60                 RTS
F8CD:  FF FF FF           DFB  $FF,$FF,$FF
F8D0:  20 82 F8  INSTDSP  JSR  INSDS1          GEN FMT, LEN BYTES
F8D3:  48                 PHA                  SAVE MNEMONIC TABLE INDEX
F8D4:  B1 3A     PRNTOP   LDA  (PCL),Y
F8D6:  20 DA FD           JSR  PRBYTE
F8D9:  A2 01              LDX  #$01            PRINT 2 BLANKS
F8DB:  20 4A F9  PRNTBL   JSR  PRBL2
F8DE:  C4 2F              CPY  LENGTH          PRINT INST (1-3 BYTES)
F8E0:  C8                 INY                  IN A 12 CHR FIELD
F8E1:  90 F1              BCC  PRNTOP
F8E3:  A2 03              LDX  #$03            CHAR COUNT FOR MNEMONIC PRINT
F8E5:  C0 04              CPY  #$04
F8E7:  90 F2              BCC  PRNTBL
F8E9:  68                 PLA                  RECOVER MNEMONIC INDEX
F8EA:  A8                 TAY
F8EB:  B9 C0 F9           LDA  MNEML,Y
F8EE:  85 2C              STA  LMNEM           FETCH 3-CHAR MNEMONIC
F8F0:  B9 00 FA           LDA  MNEMR,Y           (PACKED IN 2-BYTES)
F8F3:  85 2D              STA  RMNEM
F8F5:  A9 00     PRMN1    LDA  #$00
F8F7:  A0 05              LDY  #$05
F8F9:  06 2D     PRMN2    ASL  RMNEM           SHIFT 5 BITS OF
F8FB:  26 2C              ROL  LMNEM             CHARACTER INTO A
F8FD:  2A                 ROL  A                   (CLEARS CARRY)
F8FE:  88                 DEY
F8FF:  D0 F8              BNE  PRMN2
F901:  69 BF              ADC  #$BF            ADD "?" OFFSET
F903:  20 ED FD           JSR  COUT            OUTPUT A CHAR OF MNEM
F906:  CA                 DEX
F907:  D0 EC              BNE  PRMN1
F909:  20 48 F9           JSR  PRBLNK          OUTPUT 3 BLANKS
F90C:  A4 2F              LDY  LENGTH
F90E:  A2 06              LDX  #$06            CNT FOR 6 FORMAT BITS
F910:  E0 03     PRADR1   CPX  #$03
F912:  F0 1C              BEQ  PRADR5          IF X=3 THEN ADDR.
F914:  06 2E     PRADR2   ASL  FORMAT
F916:  90 0E              BCC  PRADR3
F918:  BD B3 F9           LDA  CHAR1-1,X
F91B:  20 ED FD           JSR  COUT
F91E:  BD B9 F9           LDA  CHAR2-1,X
F921:  F0 03              BEQ  PRADR3
F923:  20 ED FD           JSR  COUT
F926:  CA        PRADR3   DEX
F927:  D0 E7              BNE  PRADR1
F929:  60                 RTS
F92A:  88        PRADR4   DEY
F92B:  30 E7              BMI  PRADR2
F92D:  20 DA FD           JSR  PRBYTE
F930:  A5 2E     PRADR5   LDA  FORMAT
F932:  C9 E8              CMP  #$E8            HANDLE REL ADR MODE
F934:  B1 3A              LDA  (PCL),Y         SPECIAL (PRINT TARGET,
F936:  90 F2              BCC  PRADR4            NOT OFFSET)
F938:  20 56 F9  RELADR   JSR  PCADJ3
F93B:  AA                 TAX                  PCL,PCH+OFFSET+1 TO A,Y
F93C:  E8                 INX
F93D:  D0 01              BNE  PRNTYX          +1 TO Y,X
F93F:  C8                 INY
F940:  98        PRNTYX   TYA
F941:  20 DA FD  PRNTAX   JSR  PRBYTE          OUTPUT TARGET ADR
F944:  8A        PRNTX    TXA                    OF BRANCH AND RETURN
F945:  4C DA FD           JMP  PRBYTE
F948:  A2 03     PRBLNK   LDX  #$03            BLANK COUNT
F94A:  A9 A0     PRBL2    LDA  #$A0            LOAD A SPACE
F94C:  20 ED FD  PRBL3    JSR  COUT            OUTPUT A BLANK
F94F:  CA                 DEX
```

```
F950:  D0 F8          BNE  PRBL2      LOOP UNTIL COUNT=0
F952:  60             RTS
F953:  38    PCADJ    SEC             0=1-BYTE,1=2-BYTE,
F954:  A5 2F PCADJ2   LDA  LENGTH       2=3-BYTE
F956:  A4 3B PCADJ3   LDY  PCH
F958:  AA             TAX             TEST DISPLACEMENT SIGN
F959:  10 01          BPL  PCADJ4       (FOR REL BRANCH)
F95B:  88             DEY             EXTEND NEG BY DECR PCH
F95C:  65 3A PCADJ4   ADC  PCL
F95E:  90 01          BCC  RTS2       PCL+LENGTH(OR DISPL)+1 TO A
F960:  C8             INY               CARRY INTO Y (PCH)
F961:  60    RTS2     RTS
             *        FMT1 BYTES:       XXXXXXY0 INSTRS
             *        IF Y=0            THEN LEFT HALF BYTE
             *        IF Y=1            THEN RIGHT HALF BYTE
             *                               (X=INDEX)
F962:  04 20 54
F965:  30 0D    FMT1  DFB  $04,$20,$54,$
F967:  80 04 90
F96A:  03 22          DFB  $80,$04,$90,$
F96C:  54 33 0D
F96F:  80 04          DFB  $54,$33,$0D,$
F971:  90 04 20
F974:  54 33          DFB  $90,$04,$20,$
F976:  0D 80 04
F979:  90 04          DFB  $0D,$80,$04,$
F97B:  20 54 3B
F97E:  0D 80          DFB  $20,$54,$3B,$
F980:  04 90 00
F983:  22 44          DFB  $04,$90,$00,$
F985:  33 0D C8
F988:  44 00          DFB  $33,$0D,$C8,$
F98A:  11 22 44
F98D:  33 0D          DFB  $11,$22,$44,$
F98F:  C8 44 A9
F992:  01 22          DFB  $C8,$44,$A9,$
F994:  44 33 0D
F997:  80 04          DFB  $44,$33,$0D,$
F999:  90 01 22
F99C:  44 33          DFB  $90,$01,$22,$
F99E:  0D 80 04
F9A1:  90             DFB  $0D,$80,$04,$
F9A2:  26 31 87
F9A5:  9A             DFB  $26,$31,$87,$ZZXXXY01 INSTR'S
F9A6:  00       FMT2  DFB  $00          ERR
F9A7:  21             DFB  $21          IMM
F9A8:  81             DFB  $81          Z-PAGE
F9A9:  82             DFB  $82          ABS
F9AA:  00             DFB  $00          IMPLIED
F9AB:  00             DFB  $00          ACCUMULATOR
F9AC:  59             DFB  $59          (ZPAG,X)
F9AD:  4D             DFB  $4D          (ZPAG),Y
F9AE:  91             DFB  $91          ZPAG,X
F9AF:  92             DFB  $92          ABS,X
F9B0:  86             DFB  $86          ABS,Y
F9B1:  4A             DFB  $4A          (ABS)
F9B2:  85             DFB  $85          ZPAG,Y
F9B3:  9D             DFB  $9D          RELATIVE
F9B4:  AC A9 AC
F9B7:  A3 A8 A4 CHAR1    ASC  ",),#($"
F9BA:  D9 00 D8
F9BD:  A4 A4 00 CHAR2    DFB  $D9,$00,$D8,$
                *CHAR2:  "Y",0,"X$$",0
                *        MNEML         IS OF FORM:
                *        (A)  XXXXX000
                *        (B)  XXXYY100
                *        (C)  1XXX1010
                *        (D)  XXXYYY10
                *        (E)  XXXYYY01
                *             (X=INDEX)
F9C0:  1C 8A 1C
F9C3:  23 5D 8B MNEML    DFB  $1C,$8A,$1C,$
F9C6:  1B A1 9D
```

```
F9C9:   8A 1D 23               DFB   $1B,$A1,$9D,$
F9CC:   9D 8B 1D
F9CF:   A1 00 29               DFB   $9D,$8B,$1D,$
F9D2:   19 AE 69
F9D5:   A8 19 23               DFB   $19,$AE,$69,$
F9D8:   24 53 1B
F9DB:   23 24 53               DFB   $24,$53,$1B,$
F9DE:   19 A1                  DFB   $19,$A1        (A) FORMAT ABOVE
F9E0:   00 1A 5B
F9E3:   5B A5 69               DFB   $00,$1A,$5B,$
F9E6:   24 24                  DFB   $24,$24        (B) FORMAT
F9E8:   AE AE A8
F9EB:   AD 29 00               DFB   $AE,$AE,$A8,$
F9EE:   7C 00                  DFB   $7C,$00        (C) FORMAT
F9F0:   15 9C 6D
F9F3:   9C A5 69               DFB   $15,$9C,$6D,$
F9F6:   29 53                  DFB   $29,$53        (D) FORMAT
F9F8:   84 13 34
F9FB:   11 A5 69               DFB   $84,$13,$34,$
F9FE:   23 A0                  DFB   $23,$A0        (E) FORMAT
FA00:   D8 62 5A
FA03:   48 26 62     MNEMR     DFB   $D8,$62,$5A,$
FA06:   94 88 54
FA09:   44 C8 54               DFB   $94,$88,$54,$
FA0C:   68 44 E8
FA0F:   94 00 B4               DFB   $68,$44,$E8,$
FA12:   08 84 74
FA15:   B4 28 6E               DFB   $08,$84,$74,$
FA18:   74 F4 CC
FA1B:   4A 72 F2               DFB   $74,$F4,$CC,$
FA1E:   A4 8A                  DFB   $A4,$8A        (A) FORMAT
FA20:   00 AA A2
FA23:   A2 74 74               DFB   $00,$AA,$A2,$
FA26:   74 72                  DFB   $74,$72        (B) FORMAT
FA28:   44 68 B2
FA2B:   32 B2 00               DFB   $44,$68,$B2,$
FA2E:   22 00                  DFB   $22,$00        (C) FORMAT
FA30:   1A 1A 26
FA33:   26 72 72               DFB   $1A,$1A,$26,$
FA36:   88 C8                  DFB   $88,$C8        (D) FORMAT
FA38:   C4 CA 26
FA3B:   48 44 44               DFB   $C4,$CA,$26,$
FA3E:   A2 C8                  DFB   $A2,$C8        (E) FORMAT
FA40:   FF FF FF               DFB   $FF,$FF,$FF
FA43:   20 D0 F8     STEP      JSR   INSTDSP        DISASSEMBLE ONE INST
FA46:   68                     PLA                    AT (PCL,H)
FA47:   85 2C                  STA   RTNL           ADJUST TO USER
FA49:   68                     PLA                    STACK. SAVE
FA4A:   85 2D                  STA   RTNH             RTN ADR.
FA4C:   A2 08                  LDX   #$08
FA4E:   BD 10 FB     XQINIT    LDA   INITBL-1,X     INIT XEQ AREA
FA51:   95 3C                  STA   XQT,X
FA53:   CA                     DEX
FA54:   D0 F8                  BNE   XQINIT
FA56:   A1 3A                  LDA   (PCL,X)        USER OPCODE BYTE
FA58:   F0 42                  BEQ   XBRK           SPECIAL IF BREAK
FA5A:   A4 2F                  LDY   LENGTH         LEN FROM DISASSEMBLY
FA5C:   C9 20                  CMP   #$20
FA5E:   F0 59                  BEQ   XJSR           HANDLE JSR, RTS, JMP,
FA60:   C9 60                  CMP   #$60             JMP ( ), RTI SPECIAL
FA62:   F0 45                  BEQ   XRTS
FA64:   C9 4C                  CMP   #$4C
FA66:   F0 5C                  BEQ   XJMP
FA68:   C9 6C                  CMP   #$6C
FA6A:   F0 59                  BEQ   XJMPAT
FA6C:   C9 40                  CMP   #$40
FA6E:   F0 35                  BEQ   XRTI
FA70:   29 1F                  AND   #$1F
FA72:   49 14                  EOR   #$14
FA74:   C9 04                  CMP   #$04           COPY USER INST TO XEQ AREA
FA76:   F0 02                  BEQ   XQ2              WITH TRAILING NOPS
FA78:   B1 3A        XQ1       LDA   (PCL),Y        CHANGE REL BRANCH
FA7A:   99 3C 00     XQ2       STA   XQTNZ,Y          DISP TO 4 FOR
```

```
FA7D:  88                           DEY                     JMP TO BRANCH OR
FA7E:  10 F8                        BPL   XQ1               NBRANCH FROM XEQ.
FA80:  20 3F FF                     JSR   RESTORE         RESTORE USER REG CONTENTS.
FA83:  4C 3C 00                     JMP   XQTNZ           XEQ USER OP FROM RAM
FA86:  85 45           IRQ          STA   ACC               (RETURN TO NBRANCH)
FA88:  68                           PLA
FA89:  48                           PHA                   **IRQ HANDLER
FA8A:  0A                           ASL   A
FA8B:  0A                           ASL   A
FA8C:  0A                           ASL   A
FA8D:  30 03                        BMI   BREAK           TEST FOR BREAK
FA8F:  6C FE 03                     JMP   (IRQLOC)        USER ROUTINE VECTOR IN RAM
FA92:  28              BREAK        PLP
FA93:  20 4C FF                     JSR   SAV1            SAVE REG'S ON BREAK
FA96:  68                           PLA                     INCLUDING PC
FA97:  85 3A                        STA   PCL
FA99:  68                           PLA
FA9A:  85 3B                        STA   PCH
FA9C:  20 82 F8        XBRK         JSR   INSDS1          PRINT USER PC.
FA9F:  20 DA FA                     JSR   RGDSP1            AND REG'S
FAA2:  4C 65 FF                     JMP   MON             GO TO MONITOR
FAA5:  18              XRTI         CLC
FAA6:  68                           PLA                   SIMULATE RTI BY EXPECTING
FAA7:  85 48                        STA   STATUS            STATUS FROM STACK, THEN RTS
FAA9:  68              XRTS         PLA                   RTS SIMULATION
FAAA:  85 3A                        STA   PCL               EXTRACT PC FROM STACK
FAAC:  68                           PLA                     AND UPDATE PC BY 1 (LEN=0)
FAAD:  85 3B           PCINC2       STA   PCH
FAAF:  A5 2F           PCINC3       LDA   LENGTH          UPDATE PC BY LEN
FAB1:  20 56 F9                     JSR   PCADJ3
FAB4:  84 3B                        STY   PCH
FAB6:  18                           CLC
FAB7:  90 14                        BCC   NEWPCL
FAB9:  18              XJSR         CLC
FABA:  20 54 F9                     JSR   PCADJ2          UPDATE PC AND PUSH
FABD:  AA                           TAX                     ONTO STACK FOR
FABE:  98                           TYA                     JSR SIMULATE
FABF:  48                           PHA
FAC0:  8A                           TXA
FAC1:  48                           PHA
FAC2:  A0 02                        LDY   #$02
FAC4:  18              XJMP         CLC
FAC5:  B1 3A           XJMPAT       LDA   (PCL),Y
FAC7:  AA                           TAX                   LOAD PC FOR JMP,
FAC8:  88                           DEY                     (JMP) SIMULATE.
FAC9:  B1 3A                        LDA   (PCL),Y
FACB:  86 3B                        STX   PCH
FACD:  85 3A           NEWPCL       STA   PCL
FACF:  B0 F3                        BCS   XJMP
FAD1:  A5 2D           RTNJMP       LDA   RTNH
FAD3:  48                           PHA
FAD4:  A5 2C                        LDA   RTNL
FAD6:  48                           PHA
FAD7:  20 8E FD        REGDSP       JSR   CROUT           DISPLAY USER REG
FADA:  A9 45           RGDSP1       LDA   #ACC              CONTENTS WITH
FADC:  85 40                        STA   A3L               LABELS
FADE:  A9 00                        LDA   #ACC/256
FAE0:  85 41                        STA   A3H
FAE2:  A2 FB                        LDX   #$FB
FAE4:  A9 A0           RDSP1        LDA   #$A0
FAE6:  20 ED FD                     JSR   COUT
FAE9:  BD 1E FA                     LDA   RTBL-$FB,X
FAEC:  20 ED FD                     JSR   COUT
FAEF:  A9 BD                        LDA   #$BD
FAF1:  20 ED FD                     JSR   COUT
FAF4:  B5 4A                        LDA   ACC+5,X
FAF6:  20 DA FD                     JSR   PRBYTE
FAF9:  E8                           INX
FAFA:  30 E8                        BMI   RDSP1
FAFC:  60                           RTS
FAFD:  18              BRANCH       CLC                   BRANCH TAKEN,
FAFE:  A0 01                        LDY   #$01              ADD LEN+2 TO PC
FB00:  B1 3A                        LDA   (PCL),Y
```

```
FB02:  20 56 F9          JSR  PCADJ3
FB05:  85 3A             STA  PCL
FB07:  98                TYA
FB08:  38                SEC
FB09:  B0 A2             BCS  PCINC2
FB0B:  20 4A FF  NBRNCH  JSR  SAVE         NORMAL RETURN AFTER
FB0E:  38                SEC                 XEQ USER OF
FB0F:  B0 9E             BCS  PCINC3       GO UPDATE PC
FB11:  EA        INITBL  NOP
FB12:  EA                NOP               DUMMY FILL FOR
FB13:  4C 0B FB          JMP  NBRNCH         XEQ AREA
FB16:  4C FD FA          JMP  BRANCH
FB19:  C1        RTBL    DFB  $C1
FB1A:  D8                DFB  $D8
FB1B:  D9                DFB  $D9
FB1C:  D0                DFB  $D0
FB1D:  D3                DFB  $D3
FB1E:  AD 70 C0  PREAD   LDA  PTRIG        TRIGGER PADDLES
FB21:  A0 00             LDY  #$00         INIT COUNT
FB23:  EA                NOP               COMPENSATE FOR 1ST COUNT
FB24:  EA                NOP
FB25:  BD 64 C0  PREAD2  LDA  PADDL0,X     COUNT Y-REG EVERY
FB28:  10 04             BPL  RTS2D          12 USEC
FB2A:  C8                INY
FB2B:  D0 F8             BNE  PREAD2         EXIT AT 255 MAX
FB2D:  88                DEY
FB2E:  60        RTS2D   RTS
FB2F:  A9 00     INIT    LDA  #$00         CLR STATUS FOR DEBUG
FB31:  85 48             STA  STATUS         SOFTWARE
FB33:  AD 56 C0          LDA  LORES
FB36:  AD 54 C0          LDA  LOWSCR       INIT VIDEO MODE
FB39:  AD 51 C0  SETTXT  LDA  TXTSET       SET FOR TEXT MODE
FB3C:  A9 00             LDA  #$00           FULL SCREEN WINDOW
FB3E:  F0 0B             BEQ  SETWND
FB40:  AD 50 C0  SETGR   LDA  TXTCLR       SET FOR GRAPHICS MODE
FB43:  AD 53 C0          LDA  MIXSET         LOWER 4 LINES AS
FB46:  20 36 F8          JSR  CLRTOP         TEXT WINDOW
FB49:  A9 14             LDA  #$14
FB4B:  85 22     SETWND  STA  WNDTOP       SET FOR 40 COL WINDOW
FB4D:  A9 00             LDA  #$00           TOP IN A-REG,
FB4F:  85 20             STA  WNDLFT         BTTM AT LINE 24
FB51:  A9 28             LDA  #$28
FB53:  85 21             STA  WNDWDTH
FB55:  A9 18             LDA  #$18
FB57:  85 23             STA  WNDBTM         VTAB TO ROW 23
FB59:  A9 17             LDA  #$17
FB5B:  85 25     TABV    STA  CV           VTABS TO ROW IN A-REG
FB5D:  4C 22 FC          JMP  VTAB
FB60:  20 A4 FB  MULPM   JSR  MD1          ABS VAL OF AC AUX
FB63:  A0 10     MUL     LDY  #$10         INDEX FOR 16 BITS
FB65:  A5 50     MUL2    LDA  ACL          ACX * AUX + XTND
FB67:  4A                LSR  A              TO AC, XTND
FB68:  90 0C             BCC  MUL4         IF NO CARRY,
FB6A:  18                CLC                 NO PARTIAL PROD.
FB6B:  A2 FE             LDX  #$FE
FB6D:  B5 54     MUL3    LDA  XTNDL+2,X    ADD MPLCND (AUX)
FB6F:  75 56             ADC  AUXL+2,X      TO PARTIAL PROD
FB71:  95 54             STA  XTNDL+2,X         (XTND).
FB73:  E8                INX
FB74:  D0 F7             BNE  MUL3
FB76:  A2 03     MUL4    LDX  #$03
FB78:  76        MUL5    DFB  #$76
FB79:  50                DFB  #$50
FB7A:  CA                DEX
FB7B:  10 FB             BPL  MUL5
FB7D:  88                DEY
FB7E:  D0 E5             BNE  MUL2
FB80:  60                RTS
FB81:  20 A4 FB  DIVPM   JSR  MD1          ABS VAL OF AC, AUX.
FB84:  A0 10     DIV     LDY  #$10         INDEX FOR 16 BITS
FB86:  06 50     DIV2    ASL  ACL
FB88:  26 51             ROL  ACH
FB8A:  26 52             ROL  XTNDL        XTND/AUX
```

```
FB8C:  26 53              ROL   XTNDH        TO AC.
FB8E:  38                 SEC
FB8F:  A5 52              LDA   XTNDL
FB91:  E5 54              SBC   AUXL        MOD TO XTND.
FB93:  AA                 TAX
FB94:  A5 53              LDA   XTNDH
FB96:  E5 55              SBC   AUXH
FB98:  90 06              BCC   DIV3
FB9A:  86 52              STX   XTNDL
FB9C:  85 53              STA   XTNDH
FB9E:  E6 50              INC   ACL
FBA0:  88        DIV3     DEY
FBA1:  D0 E3              BNE   DIV2
FBA3:  60                 RTS
FBA4:  A0 00     MD1      LDY   #$00        ABS VAL OF AC, AUX
FBA6:  84 2F              STY   SIGN          WITH RESULT SIGN
FBA8:  A2 54              LDX   #AUXL         IN LSB OF SIGN.
FBAA:  20 AF FB           JSR   MD2
FBAD:  A2 50              LDX   #ACL
FBAF:  B5 01     MD2      LDA   LOC1,X      X SPECIFIES AC OR AUX
FBB1:  10 0D              BPL   MDRTS
FBB3:  38                 SEC
FBB4:  98        MD3      TYA
FBB5:  F5 00              SBC   LOC0,X      COMPL SPECIFIED REG
FBB7:  95 00              STA   LOC0,X        IF NEG.
FBB9:  98                 TYA
FBBA:  F5 01              SBC   LOC1,X
FBBC:  95 01              STA   LOC1,X
FBBE:  E6 2F              INC   SIGN
FBC0:  60        MDRTS    RTS
FBC1:  48        BASCALC  PHA               CALC BASE ADR IN BASL,H
FBC2:  4A                 LSR   A             FOR GIVEN LINE NO.
FBC3:  29 03              AND   #$03          0<=LINE NO.<=$17
FBC5:  09 04              ORA   #$04        ARG=000ABCDE, GENERATE
FBC7:  85 29              STA   BASH          BASH=000001CD
FBC9:  68                 PLA                 AND
FBCA:  29 18              AND   #$18          BASL=EABAB000
FBCC:  90 02              BCC   BSCLC2
FBCE:  69 7F              ADC   #$7F
FBD0:  85 28     BSCLC2   STA   BASL
FBD2:  0A                 ASL   A
FBD3:  0A                 ASL   A
FBD4:  05 28              ORA   BASL
FBD6:  85 28              STA   BASL
FBD8:  60                 RTS
FBD9:  C9 87     BELL1    CMP   #$87        BELL CHAR? (CNTRL-G)
FBDB:  D0 12              BNE   RTS2B         NO, RETURN
FBDD:  A9 40              LDA   #$40        DELAY .01 SECONDS
FBDF:  20 A8 FC           JSR   WAIT
FBE2:  A0 C0              LDY   #$C0
FBE4:  A9 0C     BELL2    LDA   #$0C        TOGGLE SPEAKER AT
FBE6:  20 A8 FC           JSR   WAIT          1 KHZ FOR .1 SEC.
FBE9:  AD 30 C0           LDA   SPKR
FBEC:  88                 DEY
FBED:  D0 F5              BNE   BELL2
FBEF:  60        RTS2B    RTS
FBF0:  A4 24     STOADV   LDY   CH          CURSER H INDEX TO Y-REG
FBF2:  91 28              STA   (BASL),Y    STOR CHAR IN LINE
FBF4:  E6 24     ADVANCE  INC   CH          INCREMENT CURSER H INDEX
FBF6:  A5 24              LDA   CH            (MOVE RIGHT)
FBF8:  C5 21              CMP   WNDWDTH     BEYOND WINDOW WIDTH?
FBFA:  B0 66              BCS   CR            YES CR TO NEXT LINE
FBFC:  60        RTS3     RTS                 NO, RETURN
FBFD:  C9 A0     VIDOUT   CMP   #$A0        CONTROL CHAR?
FBFF:  B0 EF              BCS   STOADV        NO, OUTPUT IT.
FC01:  A8                 TAY               INVERSE VIDEO?
FC02:  10 EC              BPL   STOADV        YES, OUTPUT IT.
FC04:  C9 8D              CMP   #$8D        CR?
FC06:  F0 5A              BEQ   CR            YES.
FC08:  C9 8A              CMP   #$8A        LINE FEED?
FC0A:  F0 5A              BEQ   LF            IF SO, DO IT.
FC0C:  C9 88              CMP   #$88        BACK SPACE? (CNTRL-H)
FC0E:  D0 C9              BNE   BELL1         NO, CHECK FOR BELL.
```

```
FC10:  C6 24              BS       DEC  CH          DECREMENT CURSER H INDEX
FC12:  10 E8                       BPL  RTS3        IF POS, OK. ELSE MOVE UP
FC14:  A5 21                       LDA  WNDWDTH     SET CH TO WNDWDTH-1
FC16:  85 24                       STA  CH
FC18:  C6 24                       DEC  CH          (RIGHTMOST SCREEN POS)
FC1A:  A5 22              UP       LDA  WNDTOP      CURSER V INDEX
FC1C:  C5 25                       CMP  CV
FC1E:  B0 0B                       BCS  RTS4        IF TOP LINE THEN RETURN
FC20:  C6 25                       DEC  CV          DECR CURSER V-INDEX
FC22:  A5 25              VTAB     LDA  CV          GET CURSER V-INDEX
FC24:  20 C1 FB           VTABZ    JSR  BASCALC     GENERATE BASE ADDR
FC27:  65 20                       ADC  WNDLFT      ADD WINDOW LEFT INDEX
FC29:  85 28                       STA  BASL        TO BASL
FC2B:  60                 RTS4     RTS
FC2C:  49 C0              ESC1     EOR  #$C0        ESC?
FC2E:  F0 28                       BEQ  HOME          IF SO, DO HOME AND CLEAR
FC30:  69 FD                       ADC  #$FD        ESC-A OR B CHECK
FC32:  90 C0                       BCC  ADVANCE       A, ADVANCE
FC34:  F0 DA                       BEQ  BS            B, BACKSPACE
FC36:  69 FD                       ADC  #$FD        ESC-C OR D CHECK
FC38:  90 2C                       BCC  LF            C,DOWN
FC3A:  F0 DE                       BEQ  UP            D, GO UP
FC3C:  69 FD                       ADC  #$FD        ESC-E OR F CHECK
FC3E:  90 5C                       BCC  CLREOL        E, CLEAR TO END OF LINE
FC40:  D0 E9                       BNE  RTS4          NOT F, RETURN
FC42:  A4 24              CLREOP   LDY  CH          CURSOR H TO Y INDEX
FC44:  A5 25                       LDA  CV          CURSOR V TO A-REGISTER
FC46:  48                 CLEOP1   PHA              SAVE CURRENT LINE ON STK
FC47:  20 24 FC                    JSR  VTABZ       CALC BASE ADDRESS
FC4A:  20 9E FC                    JSR  CLEOLZ      CLEAR TO EOL, SET CARRY
FC4D:  A0 00                       LDY  #$00        CLEAR FROM H INDEX=0 FOR REST
FC4F:  68                          PLA              INCREMENT CURRENT LINE
FC50:  69 00                       ADC  #$00        (CARRY IS SET)
FC52:  C5 23                       CMP  WNDBTM      DONE TO BOTTOM OF WINDOW?
FC54:  90 F0                       BCC  CLEOP1        NO, KEEP CLEARING LINES
FC56:  B0 CA                       BCS  VTAB          YES, TAB TO CURRENT LINE
FC58:  A5 22              HOME     LDA  WNDTOP      INIT CURSOR V
FC5A:  85 25                       STA  CV            AND H-INDICES
FC5C:  A0 00                       LDY  #$00
FC5E:  84 24                       STY  CH          THEN CLEAR TO END OF PAGE
FC60:  F0 E4                       BEQ  CLEOP1
FC62:  A9 00              CR       LDA  #$00        CURSOR TO LEFT OF INDEX
FC64:  85 24                       STA  CH          (RET CURSOR H=0)
FC66:  E6 25              LF       INC  CV          INCR CURSOR V(DOWN 1 LINE)
FC68:  A5 25                       LDA  CV
FC6A:  C5 23                       CMP  WNDBTM      OFF SCREEN?
FC6C:  90 B6                       BCC  VTABZ         NO, SET BASE ADDR
FC6E:  C6 25                       DEC  CV          DECR CURSOR V(BACK TO BOTTOM)
FC70:  A5 22              SCROLL   LDA  WNDTOP      START AT TOP OF SCRL WNDW
FC72:  48                          PHA
FC73:  20 24 FC                    JSR  VTABZ       GENERATE BASE ADDRESS
FC76:  A5 28              SCRL1    LDA  BASL        COPY BASL,H
FC78:  85 2A                       STA  BAS2L         TO BAS2L,H
FC7A:  A5 29                       LDA  BASH
FC7C:  85 2B                       STA  BAS2H
FC7E:  A4 21                       LDY  WNDWDTH     INIT Y TO RIGHTMOST INDEX
FC80:  88                          DEY                OF SCROLLING WINDOW
FC81:  68                          PLA
FC82:  69 01                       ADC  #$01        INCR LINE NUMBER
FC84:  C5 23                       CMP  WNDBTM      DONE?
FC86:  B0 0D                       BCS  SCRL3         YES, FINISH
FC88:  48                          PHA
FC89:  20 24 FC                    JSR  VTABZ       FORM BASL,H (BASE ADDR)
FC8C:  B1 28              SCRL2    LDA  (BASL),Y    MOVE A CHR UP ON LINE
FC8E:  91 2A                       STA  (BAS2L),Y
FC90:  88                          DEY              NEXT CHAR OF LINE
FC91:  10 F9                       BPL  SCRL2
FC93:  30 E1                       BMI  SCRL1       NEXT LINE
FC95:  A0 00              SCRL3    LDY  #$00        CLEAR BOTTOM LINE
FC97:  20 9E FC                    JSR  CLEOLZ      GET BASE ADDR FOR BOTTOM LINE
FC9A:  B0 86                       BCS  VTAB        CARRY IS SET
FC9C:  A4 24              CLREOL   LDY  CH          CURSOR H INDEX
FC9E:  A9 A0              CLEOLZ   LDA  #$A0
```

```
FCA0:  91 28     CLEOL2  STA  (BASL),Y   STORE BLANKS FROM 'HERE'
FCA2:  C8                INY               TO END OF LINES (WNDWDTH)
FCA3:  C4 21             CPY  WNDWDTH
FCA5:  90 F9             BCC  CLEOL2
FCA7:  60                RTS
FCA8:  38        WAIT    SEC
FCA9:  48        WAIT2   PHA
FCAA:  E9 01     WAIT3   SBC  #$01
FCAC:  D0 FC             BNE  WAIT3      1.0204 USEC
FCAE:  68                PLA             (13+2712*A+512*A*A)
FCAF:  E9 01             SBC  #$01
FCB1:  D0 F6             BNE  WAIT2
FCB3:  60                RTS
FCB4:  E6 42     NXTA4   INC  A4L        INCR 2-BYTE A4
FCB6:  D0 02             BNE  NXTA1        AND A1
FCB8:  E6 43             INC  A4H
FCBA:  A5 3C     NXTA1   LDA  A1L        INCR 2-BYTE A1.
FCBC:  C5 3E             CMP  A2L
FCBE:  A5 3D             LDA  A1H          AND COMPARE TO A2
FCC0:  E5 3F             SBC  A2H
FCC2:  E6 3C             INC  A1L          (CARRY SET IF >=)
FCC4:  D0 02             BNE  RTS4B
FCC6:  E6 3D             INC  A1H
FCC8:  60        RTS4B   RTS
FCC9:  A0 4B     HEADR   LDY  #$4B       WRITE A*256 'LONG 1'
FCCB:  20 DB FC          JSR  ZERDLY       HALF CYCLES
FCCE:  D0 F9             BNE  HEADR        (650 USEC EACH )
FCD0:  69 FE             ADC  #$FE
FCD2:  B0 F5             BCS  HEADR      THEN A 'SHORT 0'
FCD4:  A0 21             LDY  #$21         (400 USEC)
FCD6:  20 DB FC  WRBIT   JSR  ZERDLY     WRITE TWO HALF CYCLES
FCD9:  C8                INY               OF 250 USEC ('0')
FCDA:  C8                INY               OR 500 USEC ('0')
FCDB:  88        ZERDLY  DEY
FCDC:  D0 FD             BNE  ZERDLY
FCDE:  90 05             BCC  WRTAPE     Y IS COUNT FOR
FCE0:  A0 32             LDY  #$32         TIMING LOOP
FCE2:  88        ONEDLY  DEY
FCE3:  D0 FD             BNE  ONEDLY
FCE5:  AC 20 C0  WRTAPE  LDY  TAPEOUT
FCE8:  A0 2C             LDY  #$2C
FCEA:  CA                DEX
FCEB:  60                RTS
FCEC:  A2 08     RDBYTE  LDX  #$08       8 BITS TO READ
FCEE:  48        RDBYT2  PHA             READ TWO TRANSITIONS
FCEF:  20 FA FC          JSR  RD2BIT       (FIND EDGE)
FCF2:  68                PLA
FCF3:  2A                ROL  A          NEXT BIT
FCF4:  A0 3A             LDY  #$3A       COUNT FOR SAMPLES
FCF6:  CA                DEX
FCF7:  D0 F5             BNE  RDBYT2
FCF9:  60                RTS
FCFA:  20 FD FC  RD2BIT  JSR  RDBIT
FCFD:  88        RDBIT   DEY             DECR Y UNTIL
FCFE:  AD 60 C0          LDA  TAPEIN       TAPE TRANSITION
FD01:  45 2F             EOR  LASTIN
FD03:  10 F8             BPL  RDBIT
FD05:  45 2F             EOR  LASTIN
FD07:  85 2F             STA  LASTIN
FD09:  C0 80             CPY  #$80       SET CARRY ON Y-REG.
FD0B:  60                RTS
FD0C:  A4 24     RDKEY   LDY  CH
FD0E:  B1 28             LDA  (BASL),Y   SET SCREEN TO FLASH
FD10:  48                PHA
FD11:  29 3F             AND  #$3F
FD13:  09 40             ORA  #$40
FD15:  91 28             STA  (BASL),Y
FD17:  68                PLA
FD18:  6C 38 00          JMP  (KSWL)     GO TO USER KEY-IN
FD1B:  E6 4E     KEYIN   INC  RNDL
FD1D:  D0 02             BNE  KEYIN2     INCR RND NUMBER
FD1F:  E6 4F             INC  RNDH
FD21:  2C 00 C0  KEYIN2  BIT  KBD        KEY DOWN?
```

```
FD24:  10 F5              BPL   KEYIN        LOOP
FD26:  91 28              STA   (BASL),Y   REPLACE FLASHING SCREEN
FD28:  AD 00 C0           LDA   KBD        GET KEYCODE
FD2B:  2C 10 C0           BIT   KBDSTRB    CLR KEY STROBE
FD2E:  60                 RTS
FD2F:  20 0C FD  ESC      JSR   RDKEY      GET KEYCODE
FD32:  20 2C FC           JSR   ESC1         HANDLE ESC FUNC.
FD35:  20 0C FD  RDCHAR   JSR   RDKEY      READ KEY
FD38:  C9 9B              CMP   #$9B       ESC?
FD3A:  F0 F3              BEQ   ESC          YES, DON'T RETURN
FD3C:  60                 RTS
FD3D:  A5 32     NOTCR    LDA   INVFLG
FD3F:  48                 PHA
FD40:  A9 FF              LDA   #$FF
FD42:  85 32              STA   INVFLG     ECHO USER LINE
FD44:  BD 00 02           LDA   IN,X         NON INVERSE
FD47:  20 ED FD           JSR   COUT
FD4A:  68                 PLA
FD4B:  85 32              STA   INVFLG
FD4D:  BD 00 02           LDA   IN,X
FD50:  C9 88              CMP   #$88       CHECK FOR EDIT KEYS
FD52:  F0 1D              BEQ   BCKSPC       BS, CTRL-X.
FD54:  C9 98              CMP   #$98
FD56:  F0 0A              BEQ   CANCEL
FD58:  E0 F8              CPX   #$F8       MARGIN?
FD5A:  90 03              BCC   NOTCR1
FD5C:  20 3A FF           JSR   BELL         YES, SOUND BELL
FD5F:  E8        NOTCR1   INX              ADVANCE INPUT INDEX
FD60:  D0 13              BNE   NXTCHAR
FD62:  A9 DC     CANCEL   LDA   #$DC       BACKSLASH AFTER CANCELLED LIN
FD64:  20 ED FD           JSR   COUT
FD67:  20 8E FD  GETLNZ   JSR   CROUT      OUTPUT CR
FD6A:  A5 33     GETLN    LDA   PROMPT
FD6C:  20 ED FD           JSR   COUT       OUTPUT PROMPT CHAR
FD6F:  A2 01              LDX   #$01       INIT INPUT INDEX
FD71:  8A        BCKSPC   TXA                WILL BACKSPACE TO 0
FD72:  F0 F3              BEQ   GETLNZ
FD74:  CA                 DEX
FD75:  20 35 FD  NXTCHAR  JSR   RDCHAR
FD78:  C9 95              CMP   #PICK      USE SCREEN CHAR
FD7A:  D0 02              BNE   CAPTST       FOR CTRL-U
FD7C:  B1 28              LDA   (BASL),Y
FD7E:  C9 E0     CAPTST   CMP   #$E0
FD80:  90 02              BCC   ADDINP     CONVERT TO CAPS
FD82:  29 DF              AND   #$DF
FD84:  9D 00 02  ADDINP   STA   IN,X       ADD TO INPUT BUF
FD87:  C9 8D              CMP   #$8D
FD89:  D0 B2              BNE   NOTCR
FD8B:  20 9C FC           JSR   CLREOL     CLR TO EOL IF CR
FD8E:  A9 8D     CROUT    LDA   #$8D
FD90:  D0 5B              BNE   COUT
FD92:  A4 3D     PRA1     LDY   A1H        PRINT CR,A1 IN HEX
FD94:  A6 3C              LDX   A1L
FD96:  20 8E FD  PRYX2    JSR   CROUT
FD99:  20 40 F9           JSR   PRNTYX
FD9C:  A0 00              LDY   #$00
FD9E:  A9 AD              LDA   #$AD       PRINT '-'
FDA0:  4C ED FD           JMP   COUT
FDA3:  A5 3C     XAM8     LDA   A1L
FDA5:  09 07              ORA   #$07       SET TO FINISH AT
FDA7:  85 3E              STA   A2L          MOD 8=7
FDA9:  A5 3D              LDA   A1H
FDAB:  85 3F              STA   A2H
FDAD:  A5 3C     MOD8CHK  LDA   A1L
FDAF:  29 07              AND   #$07
FDB1:  D0 03              BNE   DATAOUT
FDB3:  20 92 FD  XAM      JSR   PRA1
FDB6:  A9 A0     DATAOUT  LDA   #$A0
FDB8:  20 ED FD           JSR   COUT       OUTPUT BLANK
FDBB:  B1 3C              LDA   (A1L),Y
FDBD:  20 DA FD           JSR   PRBYTE     OUTPUT BYTE IN HEX
FDC0:  20 BA FC           JSR   NXTA1
```

```
FDC3:  90 E6              BCC  MOD8CHK      CHECK IF TIME TO,
FDC5:  60        RTS4C    RTS                 PRINT ADDR
FDC6:  4A        XAMPM    LSR  A            DETERMINE IF MON
FDC7:  90 EA              BCC  XAM            MODE IS XAM
FDC9:  4A                 LSR  A              ADD, OR SUB
FDCA:  4A                 LSR  A
FDCB:  A5 3E              LDA  A2L
FDCD:  90 02              BCC  ADD
FDCF:  49 FF              EOR  #$FF         SUB: FORM 2'S COMPLEMENT
FDD1:  65 3C     ADD      ADC  A1L
FDD3:  48                 PHA
FDD4:  A9 BD              LDA  #$BD
FDD6:  20 ED FD           JSR  COUT         PRINT '=', THEN RESULT
FDD9:  68                 PLA
FDDA:  48        PRBYTE   PHA               PRINT BYTE AS 2 HEX
FDDB:  4A                 LSR  A              DIGITS, DESTROYS A-REG
FDDC:  4A                 LSR  A
FDDD:  4A                 LSR  A
FDDE:  4A                 LSR  A
FDDF:  20 E5 FD           JSR  PRHEXZ
FDE2:  68                 PLA
FDE3:  29 0F     PRHEX    AND  #$0F         PRINT HEX DIG IN A-REG
FDE5:  09 B0     PRHEXZ   ORA  #$B0           LSB'S
FDE7:  C9 BA              CMP  #$BA
FDE9:  90 02              BCC  COUT
FDEB:  69 06              ADC  #$06
FDED:  6C 36 00  COUT     JMP  (CSWL)       VECTOR TO USER OUTPUT ROUTINE
FDF0:  C9 A0     COUT1    CMP  #$A0
FDF2:  90 02              BCC  COUTZ        DON'T OUTPUT CTRL'S INVERSE
FDF4:  25 32              AND  INVFLG       MASK WITH INVERSE FLAG
FDF6:  84 35     COUTZ    STY  YSAV1        SAV Y-REG
FDF8:  48                 PHA               SAV A-REG
FDF9:  20 FD FB           JSR  VIDOUT       OUTPUT A-REG AS ASCII
FDFC:  68                 PLA               RESTORE A-REG
FDFD:  A4 35              LDY  YSAV1          AND Y-REG
FDFF:  60                 RTS                 THEN RETURN
FE00:  C6 34     BL1      DEC  YSAV
FE02:  F0 9F              BEQ  XAM8
FE04:  CA        BLANK    DEX               BLANK TO MON
FE05:  D0 16              BNE  SETMDZ       AFTER BLANK
FE07:  C9 BA              CMP  #$BA         DATA STORE MODE?
FE09:  D0 BB              BNE  XAMPM          NO, XAM, ADD OR SUB
FE0B:  85 31     STOR     STA  MODE         KEEP IN STORE MODE
FE0D:  A5 3E              LDA  A2L
FE0F:  91 40              STA  (A3L),Y      STORE AS LOW BYTE AS (A3)
FE11:  E6 40              INC  A3L
FE13:  D0 02              BNE  RTS5         INCR A3, RETURN
FE15:  E6 41              INC  A3H
FE17:  60        RTS5     RTS
FE18:  A4 34     SETMODE  LDY  YSAV         SAVE CONVERTED ':', '+',
FE1A:  B9 FF 01           LDA  IN-1,Y         '-', '.' AS MODE.
FE1D:  85 31     SETMDZ   STA  MODE
FE1F:  60                 RTS
FE20:  A2 01     LT       LDX  #$01
FE22:  B5 3E     LT2      LDA  A2L,X        COPY A2 (2 BYTES) TO
FE24:  95 42              STA  A4L,X          A4 AND A5
FE26:  95 44              STA  A5L,X
FE28:  CA                 DEX
FE29:  10 F7              BPL  LT2
FE2B:  60                 RTS
FE2C:  B1 3C     MOVE     LDA  (A1L),Y      MOVE (A1 TO A2) TO
FE2E:  91 42              STA  (A4L),Y        (A4)
FE30:  20 B4 FC           JSR  NXTA4
FE33:  90 F7              BCC  MOVE
FE35:  60                 RTS
FE36:  B1 3C     VFY      LDA  (A1L),Y      VERIFY (A1 TO A2) WITH
FE38:  D1 42              CMP  (A4L),Y        (A4)
FE3A:  F0 1C              BEQ  VFYOK
FE3C:  20 92 FD           JSR  PRA1
FE3F:  B1 3C              LDA  (A1L),Y
FE41:  20 DA FD           JSR  PRBYTE
FE44:  A9 A0              LDA  #$A0
FE46:  20 ED FD           JSR  COUT
```

```
FE49:  A9 A8                 LDA  #$A8
FE4B:  20 ED FD              JSR  COUT
FE4E:  B1 42                 LDA  (A4L),Y
FE50:  20 DA FD              JSR  PRBYTE
FE53:  A9 A9                 LDA  #$A9
FE55:  20 ED FD              JSR  COUT
FE58:  20 B4 FC  VFYOK       JSR  NXTA4
FE5B:  90 D9                 BCC  VFY
FE5D:  60                    RTS
FE5E:  20 75 FE  LIST        JSR  A1PC         MOVE A1 (2 BYTES) TO
FE61:  A9 14                 LDA  #$14           PC IF SPEC'D AND
FE63:  48        LIST2       PHA                 DISSEMBLE 20 INSTRS
FE64:  20 D0 F8              JSR  INSTDSP
FE67:  20 53 F9              JSR  PCADJ        ADJUST PC EACH INSTR
FE6A:  85 3A                 STA  PCL
FE6C:  84 3B                 STY  PCH
FE6E:  68                    PLA
FE6F:  38                    SEC
FE70:  E9 01                 SBC  #$01         NEXT OF 20 INSTRS
FE72:  D0 EF                 BNE  LIST2
FE74:  60                    RTS
FE75:  8A        A1PC        TXA               IF USER SPEC'D ADR
FE76:  F0 07                 BEQ  A1PCRTS        COPY FROM A1 TO PC
FE78:  B5 3C     A1PCLP      LDA  A1L,X
FE7A:  95 3A                 STA  PCL,X
FE7C:  CA                    DEX
FE7D:  10 F9                 BPL  A1PCLP
FE7F:  60        A1PCRTS     RTS
FE80:  A0 3F     SETINV      LDY  #$3F         SET FOR INVERSE VID
FE82:  D0 02                 BNE  SETIFLG        VIA COUT1
FE84:  A0 FF     SETNORM     LDY  #$FF         SET FOR NORMAL VID
FE86:  84 32     SETIFLG     STY  INVFLG
FE88:  60                    RTS
FE89:  A9 00     SETKBD      LDA  #$00         SIMULATE PORT #0 INPUT
FE8B:  85 3E     INPORT      STA  A2L            SPECIFIED (KEYIN ROUTINE)
FE8D:  A2 38     INPRT       LDX  #KSWL
FE8F:  A0 1B                 LDY  #KEYIN
FE91:  D0 08                 BNE  IOPRT
FE93:  A9 00     SETVID      LDA  #$00         SIMULATE PORT #0 OUTPUT
FE95:  85 3E     OUTPORT     STA  A2L            SPECIFIED (COUT1 ROUTINE)
FE97:  A2 36     OUTPRT      LDX  #CSWL
FE99:  A0 F0                 LDY  #COUT1
FE9B:  A5 3E     IOPRT       LDA  A2L          SET RAM IN/OUT VECTORS
FE9D:  29 0F                 AND  #$0F
FE9F:  F0 06                 BEQ  IOPRT1
FEA1:  09 C0                 ORA  #IOADR/256
FEA3:  A0 00                 LDY  #$00
FEA5:  F0 02                 BEQ  IOPRT2
FEA7:  A9 FD     IOPRT1      LDA  #COUT1/256
FEA9:  94 00     IOPRT2      STY  LOC0,X
FEAB:  95 01                 STA  LOC1,X
FEAD:  60                    RTS
FEAE:  EA                    NOP
FEAF:  EA                    NOP
FEB0:  4C 00 E0  XBASIC      JMP  BASIC        TO BASIC WITH SCRATCH
FEB3:  4C 03 E0  BASCONT     JMP  BASIC2       CONTINUE BASIC
FEB6:  20 75 FE  GO          JSR  A1PC         ADR TO PC IF SPEC'D
FEB9:  20 3F FF              JSR  RESTORE      RESTORE META REGS
FEBC:  6C 3A 00              JMP  (PCL)        GO TO USER SUBR
FEBF:  4C D7 FA  REGZ        JMP  REGDSP       TO REG DISPLAY
FEC2:  C6 34     TRACE       DEC  YSAV
FEC4:  20 75 FE  STEPZ       JSR  A1PC         ADR TO PC IF SPEC'D
FEC7:  4C 43 FA              JMP  STEP         TAKE ONE STEP
FECA:  4C F8 03  USR         JMP  USRADR       TO USR SUBR AT USRADR
FECD:  A9 40     WRITE       LDA  #$40
FECF:  20 C9 FC              JSR  HEADR        WRITE 10-SEC HEADER
FED2:  A0 27                 LDY  #$27
FED4:  A2 00     WR1         LDX  #$00
FED6:  41 3C                 EOR  (A1L,X)
FED8:  48                    PHA
FED9:  A1 3C                 LDA  (A1L,X)
```

```
FEDB:  20 ED FE             JSR  WRBYTE
FEDE:  20 BA FC             JSR  NXTA1
FEE1:  A0 1D                LDY  #$1D
FEE3:  68                   PLA
FEE4:  90 EE                BCC  WR1
FEE6:  A0 22                LDY  #$22
FEE8:  20 ED FE             JSR  WRBYTE
FEEB:  F0 4D                BEQ  BELL
FEED:  A2 10     WRBYTE     LDX  #$10
FEEF:  0A        WRBYT2     ASL  A
FEF0:  20 D6 FC             JSR  WRBIT
FEF3:  D0 FA                BNE  WRBYT2
FEF5:  60                   RTS
FEF6:  20 00 FE  CRMON      JSR  BL1          HANDLE CR AS BLANK
FEF9:  68                   PLA                 THEN POP STACK
FEFA:  68                   PLA                 AND RTN TO MON
FEFB:  D0 6C                BNE  MONZ
FEFD:  20 FA FC  READ       JSR  RD2BIT       FIND TAPEIN EDGE
FF00:  A9 16                LDA  #$16
FF02:  20 C9 FC             JSR  HEADR        DELAY 3.5 SECONDS
FF05:  85 2E                STA  CHKSUM       INIT CHKSUM=$FF
FF07:  20 FA FC             JSR  RD2BIT       FIND TAPEIN EDGE
FF0A:  A0 24     RD2        LDY  #$24         LOOK FOR SYNC BIT
FF0C:  20 FD FC             JSR  RDBIT          (SHORT 0)
FF0F:  B0 F9                BCS  RD2              LOOP UNTIL FOUND
FF11:  20 FD FC             JSR  RDBIT        SKIP SECOND SYNC H-CYCLE
FF14:  A0 3B                LDY  #$3B         INDEX FOR 0/1 TEST
FF16:  20 EC FC  RD3        JSR  RDBYTE       READ A BYTE
FF19:  81 3C                STA  (A1L,X)      STORE AT (A1)
FF1B:  45 2E                EOR  CHKSUM
FF1D:  85 2E                STA  CHKSUM       UPDATE RUNNING CHKSUM
FF1F:  20 BA FC             JSR  NXTA1        INCR A1, COMPARE TO A2
FF22:  A0 35                LDY  #$35         COMPENSATE 0/1 INDEX
FF24:  90 F0                BCC  RD3          LOOP UNTIL DONE
FF26:  20 EC FC             JSR  RDBYTE       READ CHKSUM BYTE
FF29:  C5 2E                CMP  CHKSUM
FF2B:  F0 0D                BEQ  BELL         GOOD, SOUND BELL AND RETURN
FF2D:  A9 C5     PRERR      LDA  #$C5
FF2F:  20 ED FD             JSR  COUT         PRINT "ERR", THEN BELL
FF32:  A9 D2                LDA  #$D2
FF34:  20 ED FD             JSR  COUT
FF37:  20 ED FD             JSR  COUT
FF3A:  A9 87     BELL       LDA  #$87         OUTPUT BELL AND RETURN
FF3C:  4C ED FD             JMP  COUT
FF3F:  A5 48     RESTORE    LDA  STATUS       RESTORE 6502 REG CONTENTS
FF41:  48                   PHA                 USED BY DEBUG SOFTWARE
FF42:  A5 45                LDA  ACC
FF44:  A6 46     RESTR1     LDX  XREG
FF46:  A4 47                LDY  YREG
FF48:  28                   PLP
FF49:  60                   RTS
FF4A:  85 45     SAVE       STA  ACC          SAVE 6502 REG CONTENTS
FF4C:  86 46     SAV1       STX  XREG
FF4E:  84 47                STY  YREG
FF50:  08                   PHP
FF51:  68                   PLA
FF52:  85 48                STA  STATUS
FF54:  BA                   TSX
FF55:  86 49                STX  SPNT
FF57:  D8                   CLD
FF58:  60                   RTS
FF59:  20 84 FE  RESET      JSR  SETNORM      SET SCREEN MODE
FF5C:  20 2F FB             JSR  INIT           AND INIT KBD/SCREEN
FF5F:  20 93 FE             JSR  SETVID         AS I/O DEV'S
FF62:  20 89 FE             JSR  SETKBD
FF65:  D8        MON        CLD               MUST SET HEX MODE!
FF66:  20 3A FF             JSR  BELL
FF69:  A9 AA     MONZ       LDA  #$AA         '*' PROMPT FOR MON
FF6B:  85 33                STA  PROMPT
FF6D:  20 67 FD             JSR  GETLNZ       READ A LINE
FF70:  20 C7 FF             JSR  ZMODE        CLEAR MON MODE, SCAN IDX
FF73:  20 A7 FF  NXTITM     JSR  GETNUM       GET ITEM, NON-HEX
FF76:  84 34                STY  YSAV           CHAR IN A-REG
```

```
FF78:  A0 17     1055          LDY  #$17        X-REG=0 IF NO HEX INPUT
FF7A:  88        1056 CHRSRCH  DEY
FF7B:  30 E8     1057          BMI  MON         NOT FOUND, GO TO MON
FF7D:  D9 CC FF  1058          CMP  CHRTBL,Y    FIND CMND CHAR IN TEL
FF80:  D0 F8     1059          BNE  CHRSRCH
FF82:  20 BE FF  1060          JSR  TCSUB       FOUND, CALL CORRESPONDING
FF85:  A4 34     1061          LDY  YSAV          SUBROUTINE
FF87:  4C 73 FF  1062          JMP  NXTITM
FF8A:  A2 03     1063 DIG      LDX  #$03
FF8C:  0A        1064          ASL  A
FF8D:  0A        1065          ASL  A           GOT HEX DIG,
FF8E:  0A        1066          ASL  A             SHIFT INTO A2
FF8F:  0A        1067          ASL  A
FF90:  0A        1068 NXTBIT   ASL  A
FF91:  26 3E     1069          ROL  A2L
FF93:  26 3F     1070          ROL  A2H
FF95:  CA        1071          DEX              LEAVE X=$FF IF DIG
FF96:  10 F8     1072          BPL  NXTBIT
FF98:  A5 31     1073 NXTBAS   LDA  MODE
FF9A:  D0 06     1074          BNE  NXTBS2      IF MODE IS ZERO
FF9C:  B5 3F     1075          LDA  A2H,X         THEN COPY A2 TO
FF9E:  95 3D     1076          STA  A1H,X         A1 AND A3
FFA0:  95 41     1077          STA  A3H,X
FFA2:  E8        1078 NXTBS2   INX
FFA3:  F0 F3     1079          BEQ  NXTBAS
FFA5:  D0 06     1080          BNE  NXTCHR
FFA7:  A2 00     1081 GETNUM   LDX  #$00        CLEAR A2
FFA9:  86 3E     1082          STX  A2L
FFAB:  86 3F     1083          STX  A2H
FFAD:  B9 00 02  1084 NXTCHR   LDA  IN,Y        GET CHAR
FFB0:  C8        1085          INY
FFB1:  49 B0     1086          EOR  #$B0
FFB3:  C9 0A     1087          CMP  #$0A
FFB5:  90 D3     1088          BCC  DIG         IF HEX DIG, THEN
FFB7:  69 88     1089          ADC  #$88
FFB9:  C9 FA     1090          CMP  #$FA
FFBB:  B0 CD     1091          BCS  DIG
FFBD:  60        1092          RTS
FFBE:  A9 FE     1093 TOSUB    LDA  #GO/256     PUSH HIGH-ORDER
FFC0:  48        1094          PHA                SUBR ADR ON STK
FFC1:  B9 E3 FF  1095          LDA  SUBTBL,Y    PUSH LOW ORDER
FFC4:  48        1096          PHA                SUBR ADR ON STK
FFC5:  A5 31     1097          LDA  MODE
FFC7:  A0 00     1098 ZMODE    LDY  #$00        CLR MODE, OLD MODE
FFC9:  84 31     1099          STY  MODE          TO A-REG
FFCB:  60        1100          RTS              GO TO SUBR VIA RTS
FFCC:  BC        1101 CHRTBL   DFB  $BC         F("CTRL-C")
FFCD:  B2        1102          DFB  $B2         F("CTRL-Y")
FFCE:  BE        1103          DFB  $BE         F("CTRL-E")
FFCF:  ED        1104          DFB  $ED         F("T")
FFD0:  EF        1105          DFB  $EF         F("V")
FFD1:  C4        1106          DFB  $C4         F("CTRL-K")
FFD2:  EC        1107          DFB  $EC         F("S")
FFD3:  A9        1108          DFB  $A9         F("CTRL-P")
FFD4:  BB        1109          DFB  $BB         F("CTRL-B")
FFD5:  A6        1110          DFB  $A6         F("-")
FFD6:  A4        1111          DFB  $A4         F("+")
FFD7:  06        1112          DFB  $06         F("M")  (F=EX-OR $B0+$89)
FFD8:  95        1113          DFB  $95         F("<")
FFD9:  07        1114          DFB  $07         F("N")
FFDA:  02        1115          DFB  $02         F("I")
FFDB:  05        1116          DFB  $05         F("L")
FFDC:  F0        1117          DFB  $F0         F("W")
FFDD:  00        1118          DFB  $00         F("G")
FFDE:  EB        1119          DFB  $EB         F("R")
FFDF:  93        1120          DFB  $93         F(":")
FFE0:  A7        1121          DFB  $A7         F(".")
FFE1:  C6        1122          DFB  $C6         F("CR")
FFE2:  99        1123          DFB  $99         F(BLANK)
FFE3:  B2        1124 SUBTBL   DFB  #BASCONT-1
FFE4:  C9        1125          DFB  #USR-1
FFE5:  BE        1126          DFB  #REGZ-1
```

```
FFE6:  C1        DFB  #TRACE-1
FFE7:  35        DFB  #VFY-1
FFE8:  8C        DFB  #INPRT-1
FFE9:  C3        DFB  #STEPZ-1
FFEA:  96        DFB  #OUTPRT-1
FFEB:  AF        DFB  #XBASIC-1
FFEC:  17        DFB  #SETMODE-1
FFED:  17        DFB  #SETMODE-1
FFEE:  2B        DFB  #MOVE-1
FFEF:  1F        DFB  #LT-1
FFF0:  83        DFB  #SETNORM-1
FFF1:  7F        DFB  #SETINV-1
FFF2:  5D        DFB  #LIST-1
FFF3:  CC        DFB  #WRITE-1
FFF4:  B5        DFB  #GO-1
FFF5:  FC        DFB  #READ-1
FFF6:  17        DFB  #SETMODE-1
FFF7:  17        DFB  #SETMODE-1
FFF8:  F5        DFB  #CRMON-1
FFF9:  03        DFB  #BLANK-1
FFFA:  FB        DFB  #NMI          NMI VECTOR
FFFB:  03        DFB  #NMI/256
FFFC:  59        DFB  #RESET        RESET VECTOR
FFFD:  FF        DFB  #RESET/256
FFFE:  86        DFB  #IRQ          IRQ VECTOR
FFFF:  FA        DFB  #IRQ/256
       XQTNZ     EQU  $3C
```

SYMBOLITAULUKKO (NUMEROJÄRJESTYKSESSÄ)

0000	LOCO
0022	WNDTOP
0026	GBASL
002A	BAS2L
002D	V2
002E	FORMAT
0030	COLOR
0034	YSAV
0038	KSWL
003C	A1L
0040	A3L
0044	A5L
0047	YREG
004F	RNDH
03F2	SOFTEV
03FB	NMI
C000	IOADR
C030	SPKR
C053	MIXSET
C057	HIRES
C05B	CLRAN1
C05F	CLRAN3
CFFF	CLRROM
F80C	RTMASK
F826	VLINEZ
F836	CLRTOP
F856	GBCALC
F87F	RTMSKZ
F8A5	ERR
F8C9	MNNDX3
F8F5	NXTCOL
F926	PRADR3
F940	PRNTYX
F94A	PRBL2
F956	PCADJ3
F9A6	FMT2
FA00	MNEMR
FA62	RESET
FAA3	NOFIX
FABA	SLOOP
FAE4	RDSP1
FB11	XLTBL
FB2E	RTS2D
FB4B	SETWND
FB6F	SETPWRC
FB97	ESCOLD
FBD0	BASCLC2
FBF0	STORADV
FC10	BS
FC2B	RTS4
FC58	HOME
FC76	SCRL1
FC9E	CLEOLZ
FCAA	WAIT3
FCC9	HEADR
FCE5	WRTAPE
FCFD	RDBIT
FD2F	ESC
FD62	CANCEL
0001	LOC1
0023	WNDBTM
0027	GBASH
002B	BAS2H
002D	RMNEM
002F	LASTIN
0031	MODE
0035	YSAV1
0039	KSWH
003D	A1H
0041	A3H
0045	A5H
0048	STATUS
0095	PICK
03F4	PWREDUP
03FE	IRQLOC
C000	KBD
C050	TXTCLR
C054	LOWSCR
C058	SETAN0
C05C	SETAN2
C060	TAPEIN
E000	BASIC
F80E	PLOT1
F828	VLINE
F838	CLRSC2
F864	SETCOL
F882	INSDS1
F8A9	GETFMT
F8D0	INSTDSP
F8F9	PRMN2
F92A	PRADR4
F941	PRNTAX
F94C	PRBL3
F95C	PCADJ4
F9B4	CHAR1
FA40	IRQ
FA6F	INITAN
FAA6	PWRUP
FAC7	NXTBYT
FAFD	PWRCON
FB19	RTBL
FB2F	INIT
FB5B	TABV
FB78	VIDWAIT
FB9B	ESCNOW
FBD9	BELL1
FBF4	ADVANCE
FC1A	UP
FC2C	ESC1
FC62	CR
FC8C	SCRL2
FCA0	CLEOL2
FCB4	NXTA4
FCD6	WRBIT
FCEC	RDBYTE
FD0C	RDKEY
FD35	RDCHAR
FD67	GETLNZ
0020	WNDLFT
0024	CH
0028	BASL
002C	H2
002E	MASK
002F	LENGTH
0032	INVFLG
0036	CSWL
003A	PCL
003E	A2L
0042	A4L
0045	ACC
0049	SPNT
0200	IN
03F5	AMPERV
0400	LINE1
C010	KBDSTRB
C051	TXTSET
C055	HISCR
C059	CLRAN0
C05D	CLRAN2
C064	PADDL0
E003	BASIC2
F819	HLINE
F831	RTS1
F83C	CLRSC3
F871	SCRN
F88C	INSDS2
F8BE	MNNDX1
F8D4	PRNTOP
F910	PRADR1
F930	PRADR5
F944	PRNTX
F953	PCADJ
F961	RTS2

```
F9BA CHAR2
FA4C BREAK
FA81 NEWMON
FAA9 SETPG3
FAD7 REGDSP
FB02 DISKID
FB1E PREAD
FB39 SETTXT
FB60 APPLEII
FB88 KBDWAIT
FBA5 ESCNEW
FBE4 BELL2
FBFC RTS3
FC22 VTAB
FC42 CLREOP
FC66 LF
FC95 SCRL3
FCA8 WAIT
FCBA NXTA1
FCDB ZERDLY
FCEE RDBYT2
FD1B KEYIN
FD3D NOTCR
FD6A GETLN
0021 WNDWDTH
0025 CV
0029 BASH
002C LMNEM
002E CHKSUM
002F SIGN
0033 PROMPT
0037 CSWH
003B PCH
003F A2H
0043 A4H
0046 XREG
004E RNDL
03F0 BRKV
03F8 USRADR
07F8 MSLOT
C020 TAPEOUT
C052 MIXCLR
C056 LORES
C05A SETAN1
C05E SETAN3
C070 PTRIG
F800 PLOT
F81C HLINE1
F832 CLRSCR
F847 GBASCALC
F879 SCRN2
F89B IEVEN
F8C2 MNNDX2
F8DB PRNTBL
F914 PRADR2
F938 RELADR
F948 PRBLNK
F954 PCADJ2
F962 FMT1
F9C0 MNEML
FA59 OLDBRK
FA9B FIXSEV
FAAB SETPLP
FADA RGDSP1
FB09 TITLE
FB25 PREAD2
FB40 SETGR
FB65 STITLE
FB94 NOWAIT
FBC1 BASCALC
FBEF RTS2B
FBFD VIDOUT
FC24 VTABZ
FC46 CLEOP1
FC70 SCROLL
FC9C CLREOL
FCA9 WAIT2
FCC8 RTS4B
FCE2 ONEDLY
FCFA RD2BIT
FD21 KEYIN2
FD5F NOTCR1
FD71 BCKSPC
FD75 NXTCHAR
FD92 PRA1
FDB3 XAM
FDD1 ADD
FDED COUT
FE04 BLANK
FE1D SETMDZ
FE36 VFY
FE75 A1PC
FE84 SETNORM
FE8D INPRT
FE9B IOPRT
FEB3 BASCONT
FEC4 STEPZ
FEED WRBYTE
FF0A RD2
FF3F RESTORE
FF59 OLDRST
FF7A CHRSRCH
FFA2 NXTBS2
FFC7 ZMODE
FD7E CAPTST
FD96 PRYX2
FDB6 DATAOUT
FDDA PRBYTE
FDF0 COUT1
FE0B STOR
FE20 LT
FE58 VFYOK
FE78 A1PCLP
FE86 SETIFLG
FE93 SETVID
FEA7 IOPRT1
FEB6 GO
FECA USR
FEEF WRBYT2
FF16 RD3
FF44 RESTR1
FF65 MON
FF8A DIG
FFA7 GETNUM
FFCC CHRTBL
FD84 ADDINP
FDA3 XAM8
FDC5 RTS4C
FDE3 PRHEX
FDF6 COUTZ
FE17 RTS5
FE22 LT2
FE5E LIST
FE7F A1PCRTS
FE89 SETKBD
FE95 OUTPORT
FEA9 IOPRT2
FEBF REGZ
FECD WRITE
FEF6 CRMON
FF2D PRERR
FF4A SAVE
FF69 MONZ
FF90 NXTBIT
FFAD NXTCHR
FFE3 SUBTBL
FD8E CROUT
FDAD MOD8CHK
FDC6 XAMPM
FDE5 PRHEXZ
FE00 BL1
FE18 SETMODE
FE2C MOVE
FE63 LIST2
FE80 SETINV
FE8B INPORT
FE97 OUTPRT
FEB0 XBASIC
FEC2 TRACE
FED4 WR1
FEFD READ
FF3A BELL
```

FF4C SAV1
FF73 NXTITM
FF98 NXTBAS
FFBE TOSUB

SYMBOLITAULUKKO (AAKKOSJÄRJESTYKSESSÄ)

003D	A1H	F956	PCADJ3	FEA7	IOPRT1
FE7F	A1PCRTS	0095	PICK	FA40	IRQ
0040	A3L	F910	PRADR1	FD1B	KEYIN
0044	A5L	F930	PRADR5	002F	LASTIN
FBF4	ADVANCE	FDDA	PRBYTE	FE5E	LIST
002A	BAS2L	FDE3	PRHEX	0001	LOC1
0029	BASH	F8DB	PRNTBL	FE20	LT
FD71	BCKSPC	0033	PROMPT	F9C0	MNEML
FE00	BL1	03F4	PWREDUP	F8C9	MNNDX3
FC10	BS	FF16	RD3	FF65	MON
F9BA	CHAR2	FD35	RDCHAR	03FB	NMI
0024	CH	FAD7	REGDSP	FB94	NOWAIT
C059	CLRANO	FF3F	RESTORE	FF90	NXTBIT
FC9C	CLREOL	004F	RNDH	FFAD	NXTCHR
F83C	CLRSC3	F87F	RTMSKZ	FF59	OLDRST
FDED	COUT	F961	RTS2	C064	PADDLO
FC62	CR	003C	A1L	F95C	PCADJ4
0025	CV	003F	A2H	F80E	PLOT1
F8A5	ERR	0043	A4H	F914	PRADR2
FB97	ESCOLD	0045	ACC	F94A	PRBL2
F9A6	FMT2	03F5	AMPERV	FB1E	PREAD
0026	GBASL	FBC1	BASCALC	FDE5	PRHEXZ
FD6A	GETLN	E000	BASIC	F8D4	PRNTOP
FCC9	HEADR	FBD9	BELL1	FD96	PRYX2
F819	HLINE	FE04	BLANK	FAA6	PWRUP
0200	IN	FD62	CANCEL	FCFD	RDBIT
F882	INSDS1	002E	CHKSUM	FD0C	RDKEY
C000	IOADR	FCA0	CLEOL2	FEBF	REGZ
03FE	IRQLOC	C05B	CLRAN1	FF44	RESTR1
C000	KBD	FC42	CLREOP	004E	RNDL
0038	KSWL	F832	CLRSCR	F831	RTS1
0400	LINE1	FDF0	COUT1	FBFC	RTS3
0000	LOC0	FEF6	CRMON	FE78	A1PCLP
FE22	LT2	FDB6	DATAOUT	003E	A2L
C053	MIXSET	FC2C	ESC1	0042	A4L
F8C2	MNNDX2	FD2F	ESC	FD84	ADDINP
FF69	MONZ	002E	FORMAT	FB60	APPLEII
FA81	NEWMON	F856	GBCALC	FBD0	BASCLC2
FD5F	NOTCR1	FFA7	GETNUM	E003	BASIC2
FF98	NXTBAS	C057	HIRES	FBE4	BELL2
FD75	NXTCHAR	FC58	HOME	FA4C	BREAK
FA59	OLDBRK	FB2F	INIT	FD7E	CAPTST
FE97	OUTPRT	F88C	INSDS2	FF7A	CHRSRCH

FC9E CLEOLZ
C05D CLRAN2
CFFF CLRROM
F836 CLRTOP
FDF6 COUTZ
0037 CSWH
FF8A DIG
FBA5 ESCNEW
FA9B FIXSEV
F847 GBASCALC
F8A9 GETFMT
FEB6 GO
C055 HISCR
F89B IEVEN
FE8B INPORT
F8D0 INSTDSP
FEA9 IOPRT2
C010 KBDSTRB
FD21 KEYIN2
002F LENGTH
FE63 LIST2
C056 LORES
002E MASK
FA00 MNEMR
FDAD MOD8CHK
FE2C MOVE
FAA3 NOFIX
FCBA NXTA1
FFA2 NXTBS2
F8F5 NXTCOL
FCE2 ONEDLY
F954 PCADJ2
003B PCH
F800 PLOT
F926 PRADR3
F94C PRBL3
FB25 PREAD2
F8F9 PRMN2
F944 PRNTX
C070 PTRIG
FCFA RD2BIT
FCEE RDBYT2
FAE4 RDSP1
F938 RELADR
FADA RGDSP1
FB19 RTBL
FBEF RTS2B
FCC8 RTS4B
FE75 A1PC
0041 A3H
0045 A5H
FDD1 ADD
002B BAS2H
FEB3 BASCONT
0028 BASL
FF3A BELL
03F0 BRKV
F9B4 CHAR1
FFCC CHRTBL
FC46 CLEOP1
C05F CLRAN3
F838 CLRSC2
0030 COLOR
FD8E CROUT
0036 CSWL
FB02 DISKID
FB9B ESCNOW
F962 FMT1
0027 GBASH
FD67 GETLNZ
002C H2
F81C HLINE1
FA6F INITAN
FE8D INPRT
0032 INVFLG
FE9B IOPRT
FB88 KBDWAIT
0039 KSWH
FC66 LF
002C LMNEM
C054 LOWSCR
C052 MIXCLR
F8BE MNNDX1
0031 MODE
07F8 MSLOT
FD3D NOTCR
FCB4 NXTA4
FAC7 NXTBYT
FF73 NXTITM
FE95 OUTPORT
F953 PCADJ
003A PCL
FD92 PRA1
F92A PRADR4
F948 PRBLNK
FF2D PRERR
F941 PRNTAX
F940 PRNTYX
FAFD PWRCON
FF0A RD2
FCEC RDBYTE
FEFD READ
FA62 RESET
002D RMNEM
F80C RTMASK
FB2E RTS2D
FDC5 RTS4C
FE17 RTS5
FC2B RTS4
FC76 SCRL1
F879 SCRN2
C05C SETAN2
FE86 SETIFLG
FE18 SETMODE
FB6F SETPWRC
002F SIGN
0049 SPNT
FE0B STOR
C060 TAPEIN
FEC2 TRACE
FECA USR
FE58 VFYOK
F828 VLINE
FCA8 WAIT
0022 WNDTOP
FEEF WRBYT2
FDA3 XAM8
FB11 XLTBL
0034 YSAV
FC8C SCRL2
FC70 SCROLL
C05E SETAN3
FE80 SETINV
FE84 SETNORM
FB39 SETTXT
FABA SLOOP
0048 STATUS
FBF0 STORADV
C020 TAPEOUT
C050 TXTCLR
03F8 USRADR
FBFD VIDOUT
FC24 VTABZ
FCAA WAIT3
0021 WNDWDTH
FEED WRBYTE
FDC6 XAMPM
0046 XREG
FCDB ZERDLY
FF4C SAV1
FC95 SCRL3
C058 SETAN0
F864 SETCOL
FE89 SETKBD
FAA9 SETPG3
FE93 SETVID
03F2 SOFTEV
FEC4 STEPZ
FFE3 SUBTBL
FB09 TITLE
C051 TXTSET
002D V2
FB78 VIDWAIT
FC22 VTAB
0023 WNDBTM
FED4 WR1

```
FECD WRITE
FDB3 XAM
0047 YREG
FFC7 ZMODE
FF4A SAVE
F871 SCRN
C05A SETAN1
FB40 SETGR
FE1D SETMDZ
FAAB SETPLP
FB4B SETWND
C030 SPKR
FB65 STITLE
FB5B TABV
FFBE TOSUB
FC1A UP
FE36 VFY
F826 VLINEZ
FCA9 WAIT2
0020 WNDLFT
FCD6 WRBIT
FCE5 WRTAPE
FEB0 XBASIC
0035 YSAV1

SYMBOL TABLE SIZE
2589    BYTES USED
2531    BYTES REMAINING

SLIST 4A
```

KIRJALLISUUSLUETTELO

Tässä muutamia julkaisuja, jotka saattavat tuntua kiinnostavilta:

Synertek/MOS Technology 6500 Programming Manual

Tämä käsikirja on johdatus MC6502-mikroprosessorin konekieliseen ohjelmointiin. Se kuvaa Applen mikroprosessorin konekielisen operoinnin äärimmäisen pikkutarkasti. Se ei kuitenkaan sisällä mainittavaa tietoa Applesta.
Kirjaa on saatavana Applelta. Tilausnumero on A2L0003.

Synertek/MOS Technology 6500 Hardware Manual

Tämä käsikirja sisältää yksityiskohtaisen kuvauksen Applen 6502-mikroprosessorin sisäisistä toiminnoista. Siinä on paljon tietoa mikroprosessorin suhteesta ulkoisiin laitteisiin, joista jotkut kuuluvat Appleen.
Myös tätä kirjaa on saatavissa Applelta. Tilausnumero on A2L0002.

Apple II Monitor Peeled

Tämä kirja sisältää perusteellisen kuvauksen Applen alkuperäisen Monitor ROMin alirutiineista.
Kirjaa on saatavilla tekijältä:

William E. Dougherty
14349 San Jose Street
Los Angeles, CA 91345 USA

Programming the 6502

Tämä Rodnay Zaksin kirjoittama kirja on erinomainen opas Applen 6502-mikroprosessorin kone- ja assembler-kieliseen ohjelmointiin.
Kirjaa on saatavissa yhtiöltä Sybex Incorporated, 2020 Milvia, Perkeley, CA 94704, USA. Sitä pitäisi olla saatavissa myös paikallisesta tietokonealan liikkeestä ja kirjakaupasta. Tilausnumero on C202.

6502 Applications

Tämä Rodnay Zaksin kirjoittama kirja kuvailee useita Applen 6502-mikroprosessorin sovellutuksia.
Myös tätä kirjaa on saatavana Sybexiltä. Tilausnumero on D302.

System Description: The Apple II

Tämä artikkeli, jonka on kirjoittanut Apple-tietokoneiden suunnittelija Steve Wozniak, kuvailee Apple II:n perusrakennetta ja -toimintoja.

Tämä artikkeli julkaistiin alun perin BYTE-lehden toukokuun numerossa 1977, ja sitä on saatavana BYTE Publications Inc.'lta, Peterborough NH 30458, USA.

SWEET 16: The 6502 Dream Machine

Tämä artikkeli, joka on myös Steve Wozniakin käsialaa, kuvailee tulkitsevaa SWEET 16 -konekieltä, joka sijaitsee Apple Integer BASIC ROMissa.
Artikkeli ilmestyi BYTE-lehden lokakuun numerossa 1977 ja sitä on saatavana BYTE Publications Inc'iltä, Peterborough, NH 30458, USA.

More Colors for Your Apple

Tämä Allen Watson III:n kirjoittama artikkeli kuvailee yksityiskohtaisesti Applen tarkkuusgrafiikkatilaa. Se sisältää myös Steve Wozniakin, Applen suunnittelijan, vastauksen, jossa kerrotaan muunnoksesta, jolla Applen Revision Ø:aan voidaan lisätä kaksi ylimääräistä väriä, jotka ovat käytössä Revision 1 -levyllä.
Artikkeli ilmestyi BYTE-lehden kesäkuun numerossa 1979. Tilausosoite: BYTE Publications, Inc, Peterborough, NH 30458, USA.

Call Apple (Apple Puget Sound Program Library Exchange)

Tämä on eräs suurimmista Applen käyttäjien tiedotuslehdistä. Tarkempaa tietoa saa kirjoittamalla osoitteeseen:

Apple Puget Sound Program Library Exchange
6708 39th Ave. Southwest
Seattle, Wash., 98136
USA

The Cider Press

Tämä on erään toisen kerhon tiedotuslehti. Tarkempaa tietoa saa osoitteesta:

The Cider Press
c/o The Apple Core of San Francisco
Box 4816
San Francisco, CA 94101
USA

HAKEMISTO

HAKEMISTO

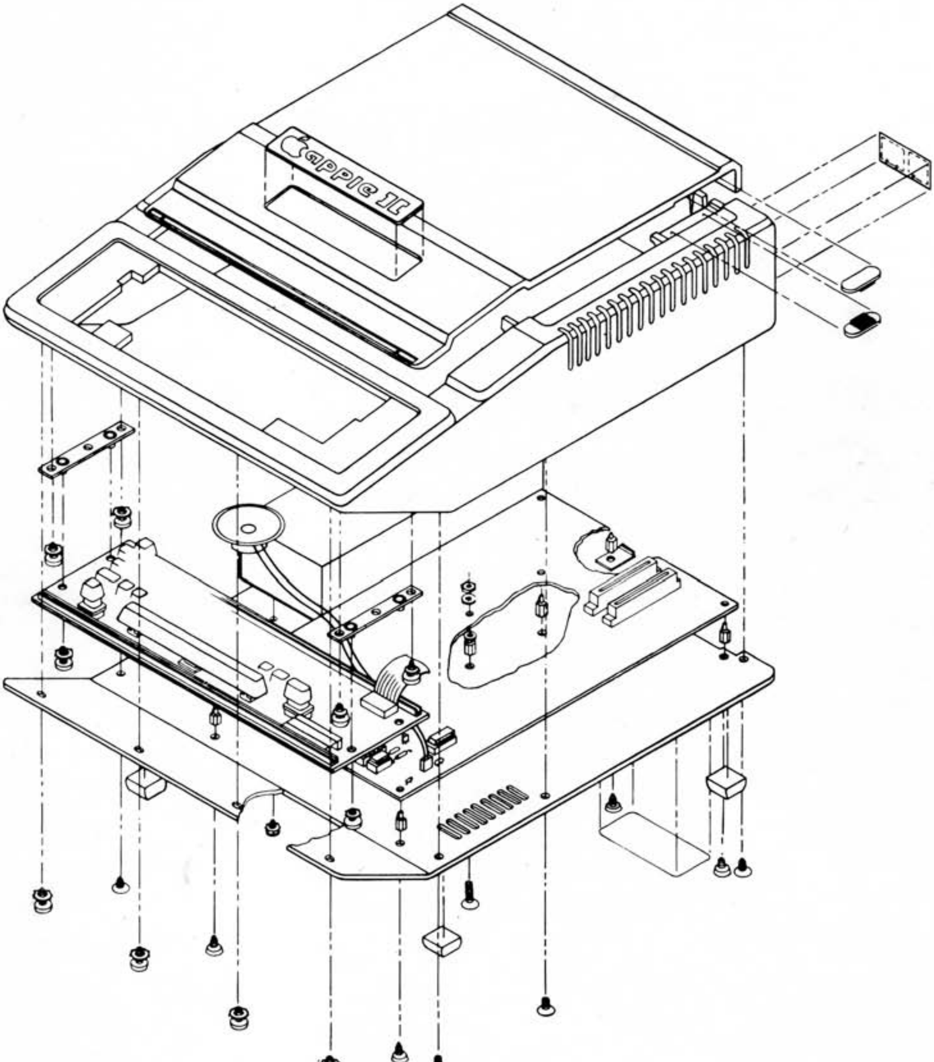
— N —

— O —

HAKEMISTO, KUVAT

HAKEMISTO, VALOKUVAT

HAKEMISTO, TAULUKOT



apple computer inc.
10260 Bandley Drive
Cupertino, California 95014